AN INVESTIGATION OF DISTRIBUTED COMMUNICATIONS SYSTEMS AND THEIR POTENTIAL APPLICATIONS TO THE COMMAND AND CONTROL STRUCTURE OF THE MARINE CORPS

By Edmund Andrew Lucke

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AN INVESTIGATION OF DISTRIBUTED COMMUNICATIONS SYSTEMS AND THEIR POTENTIAL APPLICATIONS TO COMMAND AND CONTROL STRUCTURE OF THE MARINE CORPS

by

Edmund Andrew Lucke

December 1979

Thesis Advisor: J. M. Wozencraft

Approved for public release; distribution unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): An Investigation of Distributed Communications Systems and Their Potential Applications to the Command and Control Structure of the Marine Corps	5. TYPE OF REPORT & PERIOD COVERED: Master's Thesis; December 1979
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Edmund Andrew Lucke	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93940	12. REPORT DATE: December 1979
	13. NUMBER OF PAGES: 83
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Naval Postgraduate School, Monterey, California 93940	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Distributed Communications, Optimal Routing, Network Management, Short Path Algorithms, Routing Protocols, Network Synchronization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Marine Corps Tactical Command and Control System (MTACCS) is expected to provide increased decision making speed and power through automated processing and display of data which previously was processed manually. The landing Force Organizational Systems Study (LFOSS) has challenged Marines to examine "how Command and Control will be exercised, and what the organizational impact will be" of MTACCS. (MAR 78)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

This thesis is aimed primarily at the post MTACCS concepts of Command, Control and Communications (C3). It has been heavily influenced by the areas of concern which the acquisition of MTACCS has generated. The areas of mobility, maintainability, survivability and implementation are all areas in which distributed systems hold promise of improvement upon those provided by the hierarchically structured MTACCS architecture.

Here we examine the required architecture for a second generation automated C3 system entitled the Mobile Command Concept (MCC).

Approved for public release; distribution unlimited

An Investigation of Distributed Communications Systems and Their Potential Applications to the Command and Control Structure of the Marine Corps

by

Edmund Andrew Lucke
Captain, United States Marine Corps
B.S., United States Naval Academy, 1970

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1979

ABSTRACT

The Marine Corps Tactical Command and Control System (MTACCS) is expected to provide increased decision making speed and power through automated processing and display of data which previously was processed manually. The Landing Force Organizational Systems Study (LFOSS) has challenged Marines to examine "how Command and Control will be exercised, and what the organizational impact will be" of MTACCS. (MAR 78)

This thesis is aimed primarily at the post MTACCS concepts of Command, Control and Communications (C3). It has been heavily influenced by the areas of concern which the acquisition of MTACCS has generated. The areas of mobility, maintainability, survivability and implementation are all areas in which distributed systems hold promise of improvement upon those provided by the hierarchically structured MTACCS architecture.

Here we examine the required architecture for a second generation automated C3 system entitled the Mobile Command Concept (MCC).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

It is easy to stumble while traveling the road to knowledge. For helping hands along the way I give grateful thanks to the following friends.

To "Guido", for answering untold numbers of inane questions about Fortran and CP/CMS. To Lt. Ellen F. Roland, USN, without whose help the simulation would never have been realized. To Professor Wozencraft, who deserves most of the credit for the new concepts presented here. To Carol, Eddie, and Stephanie-Ann, without whose love and understanding none of it would have been possible.

Thank you all.

I. INTRODUCTION

A. BACKGROUND

Modern warfare has become so intense and so lethal that correct and timely decisions by the Commander are essential to victory. The necessity of making a large number of such decisions over an extended period of time appears to accurately describe the nature of the problem facing the Commander and his staff in many future conflicts. Automated aids to the decision-making process have been proposed as the best way to help the Commander achieve his stated mission. A large number of such systems will soon be arriving in the inventory of the Armed Forces of the United States. They represent the solution to the problem of how to transmit, process and display increasing volumes of digital information rapidly, accurately and securely within the traditional hierarchical Command and Control structure. This was a natural first approach to the problem: automation of routine manual functions. However, as large and more sophisticated systems came upon the scene to "aid" the Commander, it has been recognized that they may instead frustrate his success if they fail to perform as expected. It would be even more galling if they were to fail because of inability on our part to effectively operate and properly maintain them, or if the C3 structure which they support is not flexible enough to adapt to tactical needs. Unstated in these concerns is

the very real problem that these large and more capable systems may not be able to withstand the rigorous environment in which they will have to operate. Indeed their presence may make a Command Post an even higher value target. (NAV 78)

The complex set of problems surrounding C3 support to tactical commanders became more vexing when materials were published outlining a Soviet counter-C3 strategy. Radio Electronic Combat (REC) is their plan for integrating signals Intelligence (SIGINT), target acquisition, Electronic Countermeasures (ECM), Electronic Support Measures (ESM), electronically supported firepower assets, and Electronic Counter-Counter Measures (ECCM). The purpose of this integration is a planned sequence of activities that will selectively deprive Soviet opponents of control of the tactical electromagnetic environment.

REC has two basic tenets. First, it attempts to locate communication "keystones" upon which the command and control of U.S. tactical forces and weapons systems are dependent. Second, as knowledge of these "keystones" is developed by Soviet intelligence they are prioritized according to their expected impact on the battle. This is done so they may be selectively neutralized (by ECM or supporting arms). Such a strategy demands a move away from rigidly structured network architectures. (MAR 79) Chapter III discusses the implications of structured networks.

It is not the nature of technology to be a panacea; rather, the ingenious application of technology is the real

strength it brings to a problem. Thus at the heart of the long term C3 problem for the Commander is the need to find an ingenious application of the rapidly expanding technology in this area. Merely repeating previous solutions with more capable devices is not the answer, since the structure of these solutions themselves appears to be their greatest problem. Simply reoptimizing traditional solutions is not "good enough".

B. DISTRIBUTED SYSTEMS

A candidate solution is to apply the concept of distributed systems to the C3 problem in the tactical environment and to evaluate its performance against more traditionally structured systems. The actual use of distributed systems has not yet been implemented in an operational environment, but several new systems are beginning to display distributed features. The impetus for the move toward distributed systems came from the ARPANET experiment managed by the Defense Advanced Research Projects Agency (DARPA). This net is an interconnection of computers and terminal users at widely distant locations which operate in a distributed manner using leased telephone lines as transmission links. This experiment has led to many new and valuable discoveries affecting both the public and private sectors of the telecommunications and data processing arenas. This type of technology advancement holds promise for tactical systems which plan to interconnect host computer systems with cabling. However, on a broader scale systems which use a

hybrid mix of cable, radio frequency (RF) links and other transmission systems are planned. Once again DARPA is in the lead in this area as they are presently conducting a field test of ARPANET technology with the Army at Fort Bragg, N.C. (XVIII Airborne Corps), and will very shortly begin a concurrent test of the impact on Command and Control structure of a 'packet radio' data distribution system. (LAW 79)

C. PLRS

The Real Time Position Location and Reporting System (RTPLRS) is now under joint development by the Army and the Marine Corps and is to be the first operational tactical system with some "distributed" features. RTPLRS is a centrally managed network of ultra-high frequency (UHF) radios connected together by an AN/UYK-20 computer to provide ground forces with real time position location, navigation and targeting information. The central computer or Master Unit (MU) possesses global knowledge of the network routine assignments and station positions which it uses to answer queries from the network subscribers and assign them functions in support of its operation. The system uses a Time Division Multiple Access (TDMA) format to ensure orderly access to the system by all users. It employs a burst transmission (low duty cycle) mode of operation with a spread spectrum waveform combined with frequency hopping for electronic-counter-counter measures (ECCM) purposes. The distributed features evidenced by RTPLRS are a dual port adaptive routing technique which allows each node to have two paths to both the MU and its

next lower level of nodes. This dual port routing is managed by the MU using an algorithm which continuously evaluates each link assignment based upon present information for each link. The approach to link optimization is not a global one; rather, it is merely an ongoing attempt to guide the network toward a preferred state. Further each node is adaptively assigned cross-link (lateral) responsibilities to monitor other nodes within its radio line-of-sight (LOS) for time-of-arrival information to be used in range measurement calculations by the MU and for possible future dual port assignment changes. This automatic restructuring of the network in response to changes in connectivity and traffic flow patterns is a precursor of a network in which no MU is necessary and each node cooperates in a network management algorithm to allow distributed operations. (USA 77) Figure 1 depicts the PLRS employment connectivity structure.

D. ADDS

As a next step in this area the Army is investigating the possibility of extending the RTPLRS technology into a data distribution system to be known as ADDS (Army Data Distribution System). They have postulated that, by using a dual network approach or a multinet architecture, they can have many subnets of RTPLRS terminals operating under the control of a local net controller (NCU) with each of these under the control of a MU. They intend for such a system to serve a dual purpose. It will be available for standard position information and for a local data transfer network

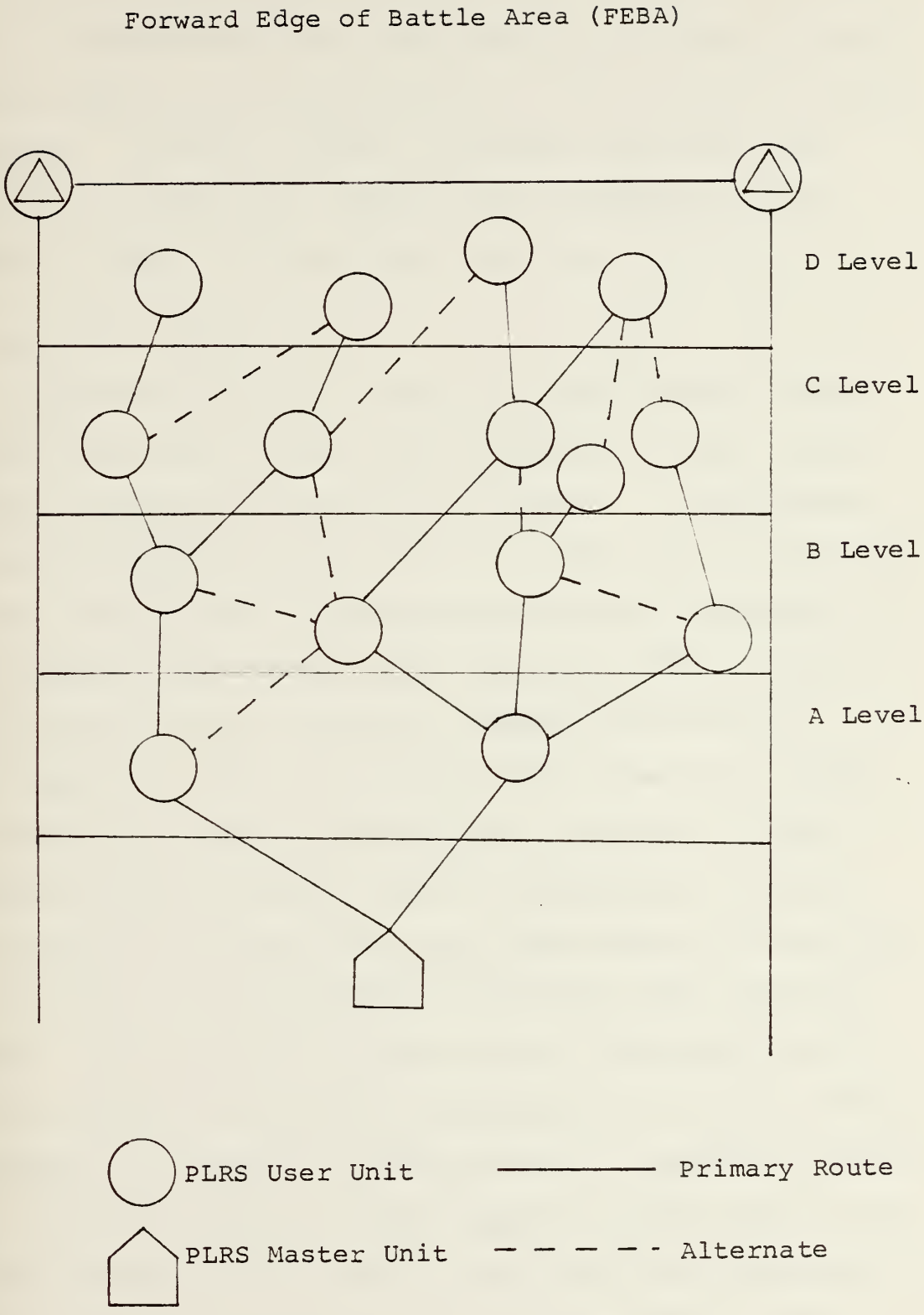


FIGURE 1. PLRS Connectivity

as well. This technique begins to lessen the dependency upon the body of knowledge stored at the MU for each transaction. The ADDS NUC's are envisioned to be flexible enough to allow operation of subnets without the MU should it become a casualty. (HUG 77) Figure 2 depicts a proposed ADDS employment connectivity, showing both the communication function as well as the position function.

E. PLRS/JTIDS HYBRID

An implementation of a similar but more capable concept is to be undertaken in the PLRS/JTID HYBRID (JTIDS: Joint Tactical Information Distribution System). In this system the existing Enhanced PLRS User Unit (EPUU) will be utilized at lower echelon units (Battalion and below) to satisfy low data transfer requirements while the basic JTIDS terminal will be used at higher echelons (Brigade and Division). (See Figure 3). This will provide high volume/high speed-of-service with improved probability of delivery to the automated systems residing there. Combinational terminals will be placed at those units with requirements to pass data to both areas of the battlefield (Artillery Fire Detection Center). As in ADDS NCU's will control PLRS subnets and a MU will oversee the PLRS/JTIDS interface and network problems. The HYBRID is to provide "communications in support of battlefield automated systems without supplanting the need for any of these systems or net radio". The HYBRID has been given an initial operational capability (IOC) target date of 1985. (USA 79) Figure 3 shows a proposed HYBRID employment concept.

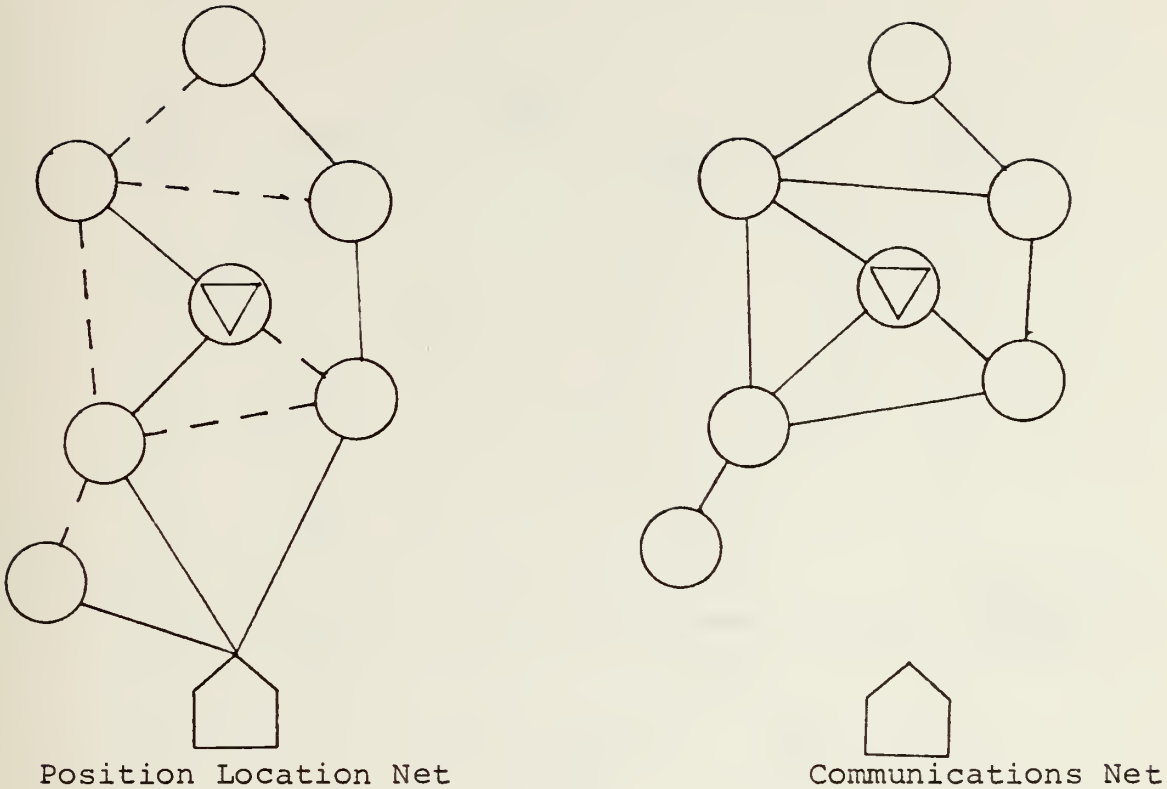


FIGURE 2. ADDS Connectivity

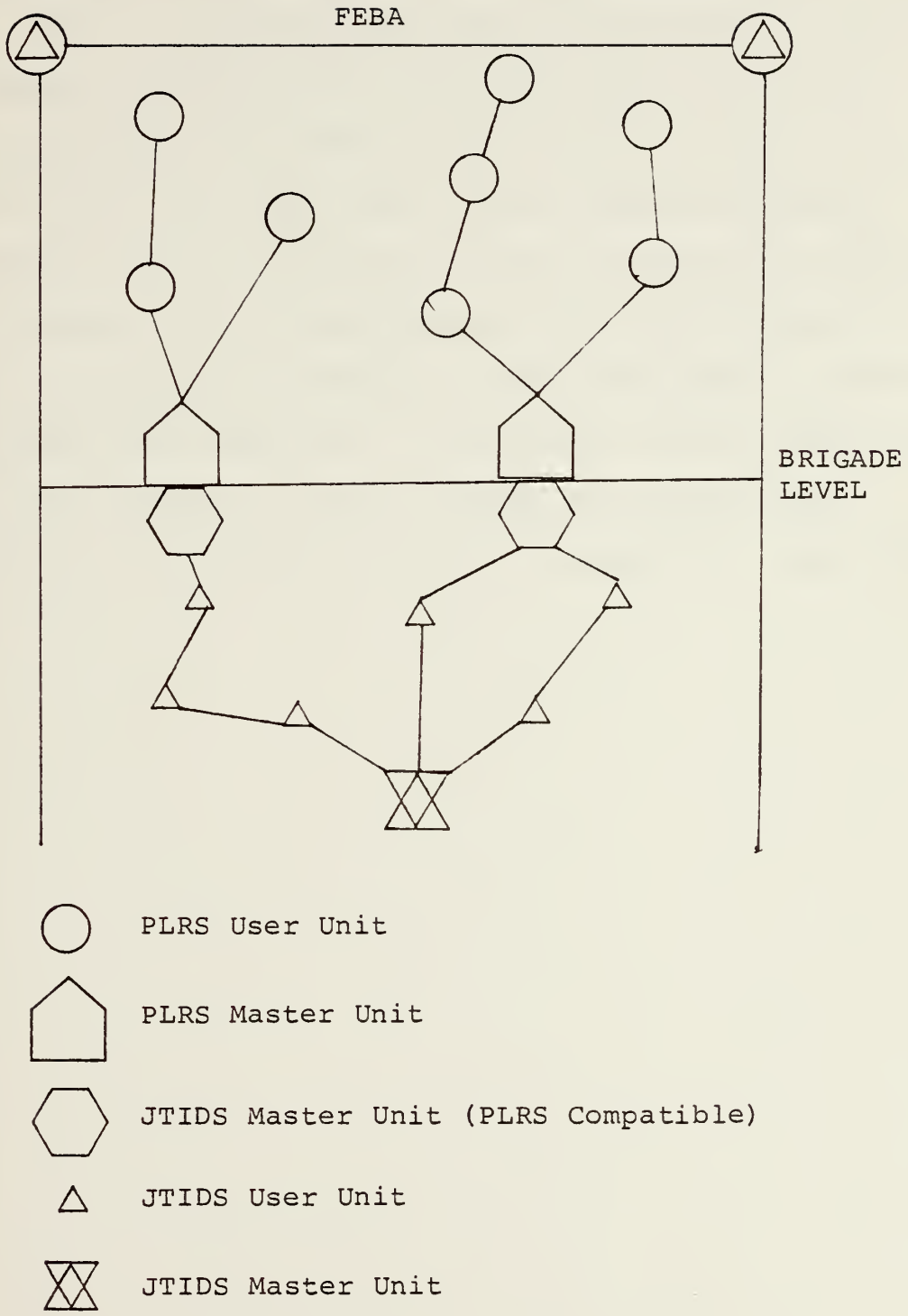


FIGURE 3. PLRS/JTIDS Hybrid Operational Concept

F. JTIDS

The final step in the presently planned implementation of semi-distributed systems is with the deployment of JTIDS itself. This is planned to be stand-alone system to provide "an advanced communications, navigation and identification (CNI) system that will serve a wide variety of users. It is planned to use a low duty cycle, spread-spectrum waveform and advanced coding techniques to provide secure jam-resistant and low probability of exploitation (LPE) CNI functions. The system will implement a Multiple Access technique", choosing from between two competitors, "to provide various levels of connectivity (access) to user elements for simultaneous distribution and receipt of digital information." (ESL 78)

II. THE NATURE OF DISTRIBUTED SYSTEMS ARCHITECTURE

A. BACKGROUND

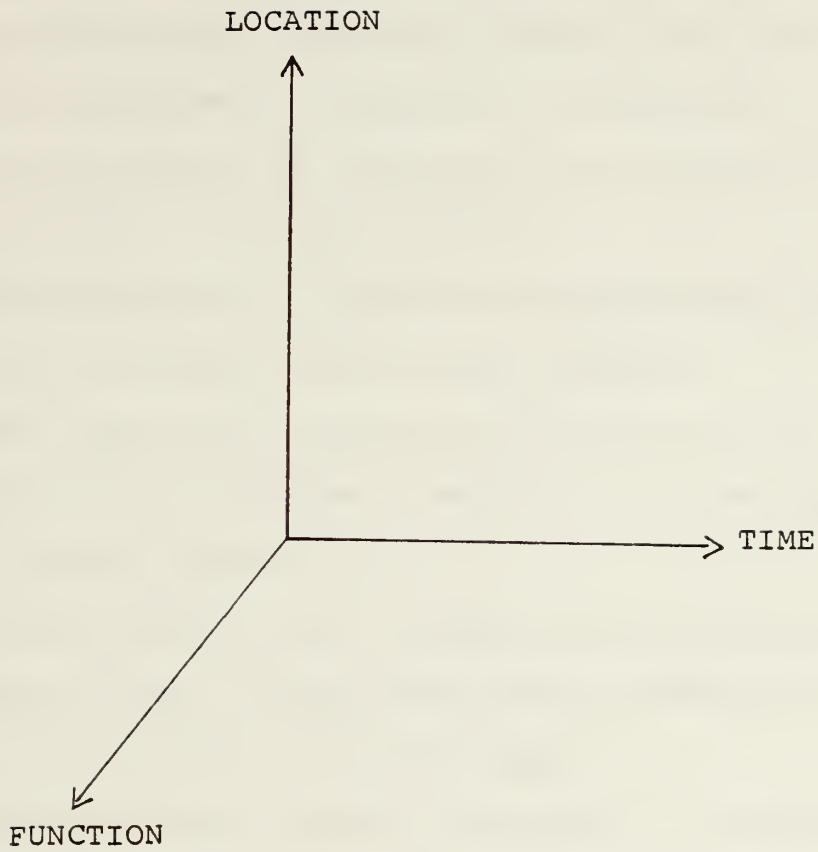
Architecture is the specification of the relationships between the parts of a system. A communications architecture for distributed systems includes the description of the formats, protocols, operational sequences, and logical structures for functions needed to achieve meaningful communications. This should be true for units having wide ranging data input/output and processing capabilities which are members of the distributed network. Additionally these units may be physically separated by large distances and interconnected by a variety of transmission media. (CYP 78)

As technological advances permit us to have computer-like capability in almost every node in a network and the cost of logic and memory circuits diminishes, we are presented with increasing opportunities to assign a higher and higher degree of functional independence in physically separate nodes. Therefore the trend toward distributed networks interconnected by a variety of transmission media, e.g., wire or fiber optic cable, RF to satellites or LOS, etc., seems inevitable. But just because it appears that we can do it is not reason enough. What can the operational commander gain by distribution and at what cost? Indeed what functions are themselves distributable and what trade-offs, if any, are there if we choose to centralize some while distributing others. Are certain of these distributable functions purely application

dependent? All these questions should be voiced early on in the architectural development of a distributed system designed for a particular application. We will now seek the answers to these and other pertinent questions that arise concerning issue--to distribute or not to distribute?

B. DISTRIBUTED FUNCTION

A function is said to be distributed if the same function can be executed at more than one node in a network, or if the function is not completely executable in a single node, and parts of that function are executed cooperatively in separate nodes. For each of these cases there are four key requirements to ensure proper execution of the function. First, within a node all information must be stored which is needed by the distributed function. Second, the decision logic must be present within the node for that function. Third, the capability within a node to execute the logic of the distributed function may be active or latent (be triggered externally). And finally, the capability to invoke the function from another node or to allocate work to that function from another node must exist. (CYP 78) Figure 4 gives a framework for the concept of the nature of distribution. Note that the dimensions on the axes of the graph are hierarchies. Note also that by distributing our function over a geographic area this puts our system in the left vertical plane only. This is a good example of a partially distributed system. By including the concept of distributing the previous functions in time we can utilize the full space described by this graph.



NOTE: Hierarchies are dimensions on axes.

FIGURE 4. The Nature of Distribution

C. TYPES OF DISTRIBUTED FUNCTIONS

Potentially there are six types of functions located in a node of a distributed network. These are related to similar functions in other nodes by a set of instructions known as internode protocols. The six classes of functions are:

1. The management of application processing.
2. The management of data that may be stored in a hierarchy of storage devices.
3. The management of communications between nodes and of the dependent source/sink devices.
4. User application program which utilize all of the first three management functions to some greater or lesser degree.
5. An intermediate set of application subsystems.
6. Input/output devices which may be dependent upon a node for their network access.

It seems clear that ideally all nodes in a distributed network should contain the first three management functions in varying degrees, with the remaining ones resident where their use is dictated by the application. Thus the evolving data-processing network can be visualized as a chain of stations of various capabilities and at various locations moving in a random fashion linked together by diverse transmission means to perform a common function. Since each node will undoubtedly have some processing power (commonality of equipment may be a key issue here), it should be the goal of the architecture to provide a common structure, incrementally if required, to meet the varying needs for communication

among the users. (CYP 78)

D. ADVANTAGES OF DISTRIBUTED FUNCTIONS

In situations where more than one central site (command post) is required or where some of these sites may be augmented by specialized or supporting systems that may be remote from these sites, the advantages of distributed function in a network may apply. Clearly this seems to be the case here as supporting systems will be sprinkled throughout the battlefield. Specifically some possible gains might be:

1. It may satisfy the need for local autonomy of function or flow control, local physical accessibility, local security or local application development.
2. It can provide greater availability to the users of the network by reducing dependence on relatively non-robust transmission links.
3. It can provide better performance from the viewpoint of the users, as it shortens the response time for processes done locally where transmission delays or cycle access rates are an important consideration.
4. It can reduce the data bandwidth required between nodes to reduce line costs and thus system costs.
5. It utilizes the more efficient properties of distinct nodes when one is more suited to a particular application than another.
6. It accommodates changes in local load patterns and applications without major disruption or expense to

the system.

7. It improves the quality of the data input leaving the source area with local aids such as edits, operator guidance and validity checks.
8. It simplifies the programming support used for a given application by tailoring the support to that limited function.

Obviously not all these advantages apply in all cases. In fact in some cases distributing functions poorly can have the opposite effect! For example, some processors will have superior response time only if their workload is low enough and they are not hamstrung by many inquiries to a central data base. Poor response time could result from a remote processor with slower speed than its host computer, depending upon their capacities and workloads. In some cases distribution of function could increase system costs for input/output equipment and personnel. This would not be the case in the MCC application, however, as it is envisioned that the nodal terminal itself would serve as the in/out port for normal tactical operations manned by operations personnel.

Thus the "optimum" distribution of function can be very application dependent as well as configuration dependent. Similarly the distribution of function should be adaptable, perhaps even adaptive, to meet changing user requirements. This diversity of trade-off makes it clear that many shades of distributed function could be considered as "optimum" for different situations. Thus an architecture for such a

distributed system must provide completely general communicability among network addressable units (nodes and the users they serve as well as external interfaces). (CYP 78) In general, delays due to increased communications requirements may also militate against a too widespread distribution of function.

III. HIERARCHICAL VERSUS DISTRIBUTED SYSTEMS

A. HIERARCHICAL SYSTEM WEAKNESSES

In order to be able to properly evaluate the strengths that distributed systems bring to the tactical C3 problem it is necessary to examine the weaknesses which hierarchical systems possess.

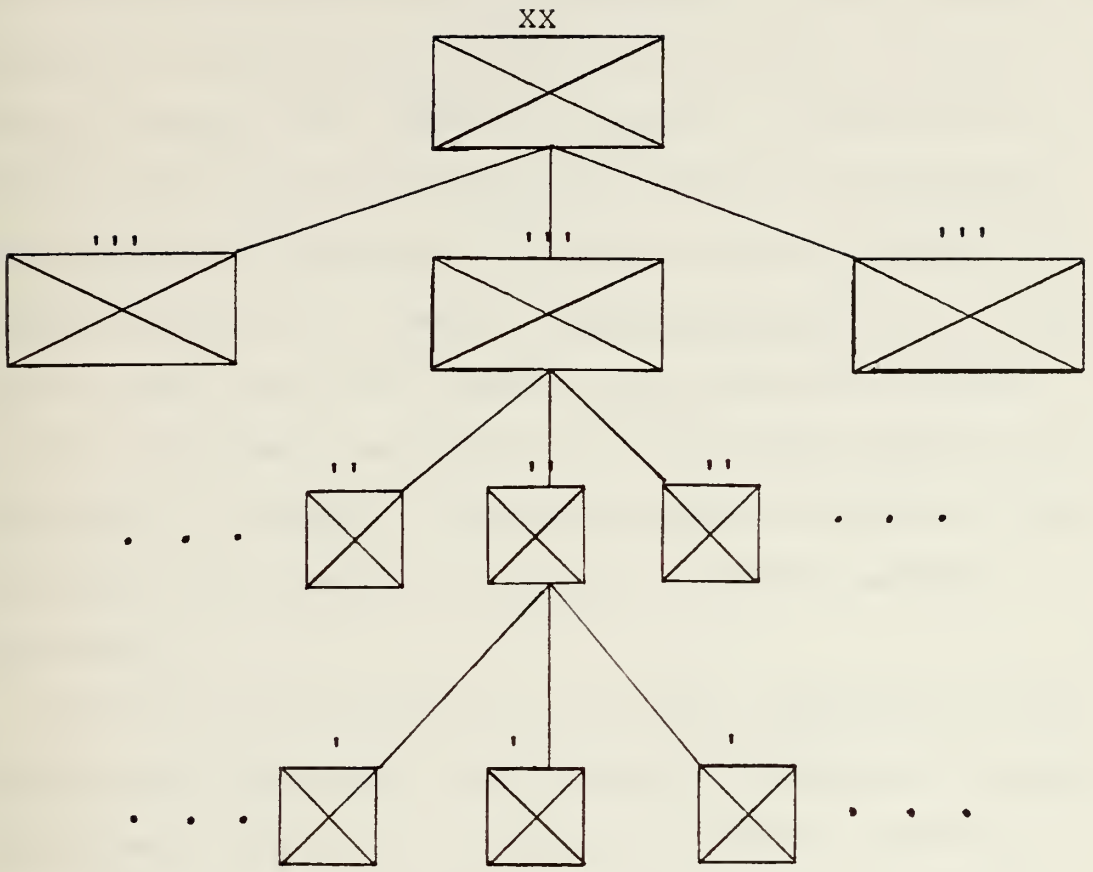
Hierarchical networks and their supporting C3 systems are in general not richly connected. That is, because they emphasize the vertical order of senior/subordinate relationships they are unable to take advantage of possible lateral connectivities which might allow more overall system robustness. This is not to say that such lateral connectivities always exist or are never used in existing tactical C3 systems rather; as a matter of course, the chain of communications follows the chain of command. This principle has been valid for as long as there have been men under arms; however, that one's command relationships should be clearly delineated and fixed is not sufficient reason to also fix communications relationships in exactly the same way. Indeed fixing these patterns in such a way gives rise to the next general area of problem.

The information mapping of such systems is a relatively simple task since the mapping function, in a mathematical sense, between the command structure and the communication links is one-to-one. Thus recovery of the communications

connectivity ensures recovery of the command relationships presently in force. Consider our system thus far. A few key nodes at each level of command, not richly connected vertically and most probably not laterally connected at all. Figure 5 shows such connectivity. Add to this picture the lack of mobility which present large C3 systems are bringing to the tactical environment and we simply fix the previous set of problems geographically for extended periods of time.

The technological sword which has generated new and more capable automated systems is a double-edged one. While on the one hand it has allowed the garnering of great processing and display power for the commander, it has required that this power be grouped close to its power source in large containers (8x8x20 feet typically). These power sources and containers are sources of infrared (IR) emissions which can be used to our detriment by IR guided munitions or enenmy sensors. This doctrinal clustering of IR targets is an unacceptable tactical situation.

Finally command post RF signatures and antenna "farms" have long been known to be undesirable side effects of our present system. New automated C3 systems do little to alleviate these problems. Remoting of antennas and emission control (EMCON) by communications and operations personnel have typically been our alternatives. There exists no doctrinal solution or support for the innovative commander who invents one to systematically overcome this problem on a broad scale.



XX = Division
''' = Regiment
'' = Battalion
' = Company

FIGURE 5. Marine Corps Command Structure (partial)

B. DISTRIBUTED SYSTEM STRENGTHS

How then do distributed systems allow us to plan to reduce the gravity of these C3 problems? From a connectivity standpoint distributed networks are designed to utilize every existing usable link within the network. Indeed while such a system does not itself imply any change in the command relationships it utilizes an adaptive process within the network structure to evaluate communications relationships such as link status and traffic flow patterns to adjust routing of messages according to a user specified set of "optimality" criteria. Thus the network becomes as richly connected as is possible in order to achieve maximum robustness.

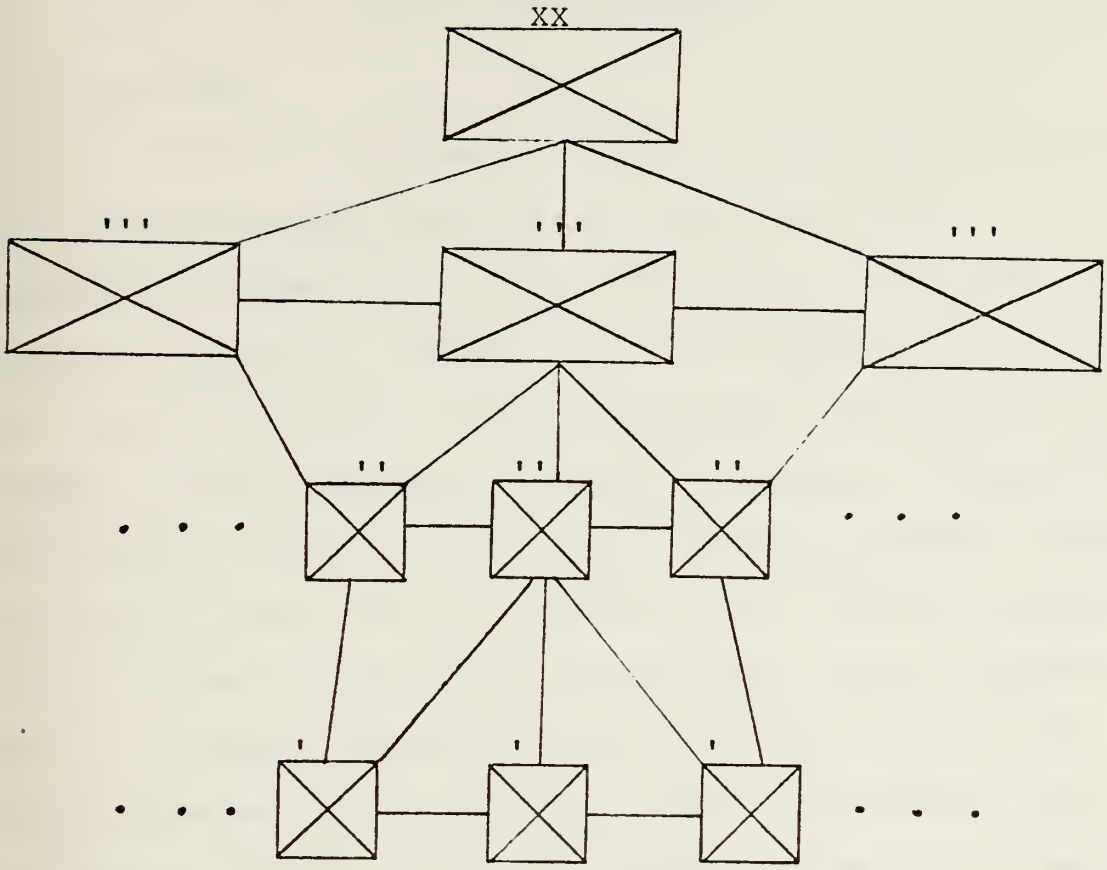
Certainly the continual or periodic generation of new routing solutions within the network in response to system requirements helps to erase the unity mapping function between command and communications structures. This feature with the ability to distribute functional capability throughout the network combine to allow us to build small and highly capable nodes (devices) which may be operated on the move in vehicles or manpack configurations. Such a system would utilize RF connectivities that changed as the nodes moved but for which the network management algorithm would compensate. This capability of mobile operations allows the commander a greater flexibility in the structuring of his command post(s) or groups beyond that presently achieved. Further it allows him a tactical flexibility seldom seen in

modern warfare. It seems implicit in distributed system design that he could disperse his staff over a wide physical area and still allow them to function effectively through the power of the network routing scheme and the use of powerful display devices. While such a concept is radically different than those in use or planned to be used it serves to illustrate the fact that no longer will commanders' resourcefulness or ingenuity be limited by their C3 system.

The battery or vehicular (jeep, truck, or tracked) power sources possible with such a system capable of mobile operations have a secondary benefit in that they reduce the IR signature of such systems. Moreover, since there are many small nodes in the network it is more difficult to correlate their value with either their position or structural placement in the network. Figure 6 shows the connectivity of such a system.

C. CONCLUSIONS

It appears then that distributed systems have much to offer the commander in a tactical environment which he cannot get from a hierarchical system. However, we should not think that any technological breakthrough alone can completely solve our problems. It remains for the Commanders of tactical units to test and evaluate candidate distributed systems to ensure that they ultimately receive a system which truly meets their needs and not one which is just a replacement for the one presently in service.



XX = Division
''' = Regiment
'' = Battalion
' = Company

FIGURE 6. Possible Distributed Communications Linkage

IV. MOBILE COMMAND CONCEPT

A. BACKGROUND

As a result of a mission analysis conducted at the Naval Ocean Systems Center (NOSC) it was determined that a new post MTACCS concept for command and control capability within the Marine Corps was needed. The long leadtime involved in the Defense procurement cycle exacerbates the need for this concept to be formulated as early as possible. Conceptually the system(s) should be capable of mounted operation riding in LVTX's, jeeps, trucks and helicopters. Certain subsystems should be man transportable without major loss of function. That is, manpack components should be able to access the overall system with only a lower level of capability than the mobile systems, not with a complete absence of capability.

Some of the initial design goals for these projected systems were that they should be totally reliable and not require field maintenance for a specified period of time, e.g., 30 to 60 days. Field maintenance in this context is anything other than a one for one replacement at the battalion level or lower. Certainly a module or "black box" concept at higher echelons is acceptable but the emphasis in this area is to relieve the tactical commander of the maintenance burden. The loss of any subsystem or component should not cause a catastrophic failure of the system. This concept of "soft fail" or graceful degradation in the face of battle damage is one which should be universally applied

to connectivity of these systems as well as to their functional capability. Finally the system(s) should be designed in such a way that the family of equipments used at battalion level form a building block of the regimental system and so on. (NAV 78)

B. PROPOSED IMPLEMENTATION

Doctrinally the Marine Corps is probably the most adaptive ground combat organization known today. The scope of the missions it is prepared to execute around the world is impressive. One reason for this wide range of capability is the way in which both its personnel and leaders are trained. "Use the 'book' when it applies and rewrite it when it doesn't." This is a phrase often heard in many a Marine training exercise. This adaptability in the face of a new situation must be called upon again to handle this "new situation".

While present doctrine is flexible the command and control systems which exists to support it are not. Rather it is constrained by the nature and limited flexibility of present and planned systems. This was not by design but was an outgrowth of the previously addressed nature of hierarchical C3 architectures. Therefore, the following proposal for a distributed C3 architectural implementation is presented as a "straw man" to begin the inevitable dialogue necessary to achieve the realization of such technology in support of the post MTACCS C3 systems in the Marine Corps.

The basic building block of a distributed network is a

nodal device which is capable of simultaneously housing communications (receive/transmit) functions, computer (processing) functions and (in this application) visual display functions. In the manpack realization such a device would be capable of full network nodal status without the need for local storage of all applications programs resident within the network. Those routines required but not resident would be invoked from other nodes. For example, if a battalion level staff officer needed a report format to generate a daily report to his senior headquarters and as a result of operational requirements was using a smaller and less powerful device while he was enroute to a briefing, he could call up the required format from within the network by asking his staff section terminal at the headquarters to forward it to him.

A battalion commander could utilize this capability to allow his staff to be subdivided, with himself, the executive officer and the operations officer as command group tactical officers (TACO's). Each group could be mounted and capable of assuming command for periods of time should a continuous combat situation arise or if the senior group were destroyed. In such a dispersal of staff functions the C.O., upon getting a new mission from his regimental headquarters, could send the situation map to the Operations Officer with a note on the new mission and a request for a plan. The S-3 would send back the map annotated with his plan and assignments. The C.O. would approve or modify it and send it off to the

Company C.O.'s.

At a more senior level where more powerful nodal devices might be required the use of "add on" on-line storage could be accomplished in a variety of ways to allow more careful analysis of data and record keeping. This storage also allows devices access to a wide range of data without overburdening the system with requests to recall it from other locations.

As might be expected visual displays of liquid crystal or CRT type will play an important role in any such system. These will enable the commander to "see" large amounts of data meaningfully displayed. They should help all key command and staff personnel to actually reduce data transfer requirements to those necessary to doing the job at hand. (One picture if worth a thousand words). Such action is vital to the success of any system but even more so in a distributed C3 system. Here the ability to transfer more data (i.e., more bandwidth) does not necessarily carry with it the need to do so. Needless overloading of the system with non-critical data merely slows or masks vital information. Although such a system is capable of both prioritizing messages and routing high priority traffic around low priority nodal blockages, these features should not be used to circumvent a real value inherent in automation. Reduction of traffic to the required messages by the use of good displays and selective use of preformatted messages is a desirable by-product of this implementation. As a first step

implementation. As a first step in this area the Army has recently published a Corps Information Flow (CIF) document which addresses this question of reducing overall data transfer requirements by identifying messages needed most in combat. These messages are broken down and formatted in matrices for ease of digital transmission. (CAC 79)

In a distributed system context the operation of a command group at any level is essentially the same whether they are configured in a traditional command post arrangement or organized in teams of command and staff personnel mounted on a variety of platforms dispersed for protection against enemy attack. The power of the network and its ability to adapt to changes allows them the flexibility to operate just as effectively even though separated. Such tactical flexibility has rarely been placed in the hands of a commander. No longer need he be needlessly constrained by his C3 system to traditional solutions to emerging problems. His personal ingenuity will be rewarded by a system which can implement his creative solution almost immediately.

More permanent installations such as Marine Air Wing C3 facilities will be able to utilize more traditional linkage concepts such as Amphibious Assault Cable (26 twisted pair) or fiber optic cable in the future. The interface of these links with the RF structure of other networks will be a major design goal of such a distributed system to ensure maximum robustness throughout. This is presently an area of high technical risk and as such requires much integrated planning and study.

V. KEY TECHNICAL QUESTIONS

A. NETWORK SYNCHRONIZATION

1. Background

To use distributed systems as a viable framework for a C3 architecture there are several technical considerations. Among these the first two which must be clearly demonstrated are those of network synchronization under any and all conditions and the ability to manage a volatile network in a distributed fashion. Without either of these capabilities distributed operation of a synchronous network is not possible.

The synchronous operation which has been implied in the preceeding pages is an assumption not easily fulfilled in the distributed network context. While the question of synchronizations of various types (bit, word, carrier, etc.) have been rigorously addressed, each implies a building block approach to be extended throughout the network. A distributed system also needs such properties. The area in which work needs to be done for distributed systems is the area of overall network synchronization. Is there a way to bring all the clocks in the network to agree on a common time? Is there an optimum approach for arbitrary networks which converges to a minimum number of transmissions necessary to accomplish this result? Can this be done without a time-out in network operations?

One recent development in this area (FIN 79) presented

a protocol having the effect of synchronizing an entire network despite arbitrary finite delays. This protocol has several other features dealing with guaranteeing no lost or duplicate packets in such a network. These "protocols do require that the whole network be brought down (time out) for the duration of the resynch procedure." The author notes cogently that this "may be too high a price to pay for the failsafe properties in a practical network." In MCC it is too high a price.

For a given network which is either completely or partially connected and operating standard synchronization procedures appear applicable. As a new member joins the network he aligns his clock to the network standard, or if he is an active member who senses his clock drifting he adjusts it upon receipt of a TODAD (Time of Day and Date) transmission. Note that implicit in both of these cases is the fact that there is an already accepted "absolute" time standard for the operating network. The source of such a standard in hierarchical structures like PLRS is the MU. In distributed networks there is no easy solution to this question. This is even more perplexing if the network is not operational and attempts to "cold start". Two cases appear viable for analysis. First: a distributed network with arbitrary connectivity operating synchronously. Second: a distributed network with arbitrary and temporarily unknown connectivity not operating. Such failure on a global network scale is envisioned as perhaps the result of a nuclear air burst over the Amphibious Operations Area.

2. Two Cases

For the first case it may reasonably be assumed that, in our tactical scenario, an operating system would have access to either a broadcast time standard (WWV) or for remote operations that nodes measure link propagation times and then pass time-of-day information from node to node. Thereafter, tracking of the carrier phase will allow all clocks to stay in synch.

For the second case, however, very little may be assumed except that all clocks are set to a different time. The remaining network connectivities, for the failed net case, are unknown by all remaining operational nodes. Thus reconstruction of the network and synchronization seem to form a difficult pair of problems requiring a simultaneous solution. If a solution to them could be found it would also be viable in case one, which is clearly less difficult.

3. Procedure for Network Synchronization

a. General

Let all nodes in the network be labeled alphabetically with the procedure for doing so being arbitrary. Let the synchronization algorithm proceed in discrete time. That is, let each operation be (Δ t) seconds after the receipt of messages which will require the operation to take place. e.g., if (Δ t) were one millisecond this would remove all propagation factors (delays) of less than one hundred and eighty six (186) miles. Finally let each node have a counter which will begin counting each (Δ t) after

that node begins, or participates initially in, the procedure. Thus a node is identified as 'A1' if it is node 'A' and this is an initial transmission. Similarly if a relaying node is 'R' and has not yet participated or initiated any TODAD's, upon receiving 'A1' it, 'R', would relay 'A2' signifying that it was in synch with Pseudo Master Clock (PMC) 'A' and this was a level '2' relay of that information. This counter is reset upon completion of the procedure or on receipt of an operational message. Such receipt would indicate operational net status, at least locally, and any further adjustments to clocks would be simply that, adjustments.

As the nodes in a network attempt to reconstitute themselves they will begin to transmit in the broadcast mode to any other node who can hear them. Such transmissions will be TODAD information and will advise the recipient by their content that it is receiving a K'th level (iteration) transmission of such information. Whenever a node receives a TODAD identified as level k, it relabels it as a level k+1 and retransmits it to all other neighbors. Any TODAD which might be received subsequently with a lower level identification is ignored. That is, the highest count wins when TODAD's collide at nodes. This prevents nodes from entering a race track or tail chase condition. If two transmission collide with equal counter numbers then arbitrarily let the low alphabetical one be dominant. This prevents unsolvable dilemmas but does not inhibit the first node coming on the

air from gaining network synch the quickest.

Thus in the simplest case the initial TODAD message proceeds outward through the network and once completed all clocks are in synch. A distributed network management algorithm could now begin to adaptively route messages through the network as in case one. Actually no notification of global acceptance of the pseudo master (PMC) is required as local (node to node) acceptance can begin the net management process and adjustments to PMC will not "undo" previously synchronized nodes nor are time-outs required to accomplish this. (see b. below).

Several other cases of interest are included in the PMC relay procedure. For example, what happens when two or more stations come up simultaneously and begin level one TODAD transmissions in adjacent or separate parts of the network? For the adjacent case the decisions needed are as previously stated: the subsequent level two relay will inform the competing nodes of the winner. At widely separated distances within the network when two TODADS collide near the center with different levels of relay the highest level of relay is dominant having, by implication, already synchronized more nodes. The continual relay of the higher level TODAD will be heard by the losing node and promulgated as an adjustment down to his previously synched neighbors. Should two TODADS meet in the center of the network with equal levels of relay then rules are again applied with the subsequent changes propagating back as before.

b. Two Station Operation

As a subset of the foregoing procedure the actual actions necessary for nodal timing were developed. If the receiver at each node is of a matched filter type and a spread spectrum waveform is employed then the following is applicable.

Consider two stations A and B needing to minimize the error ε between their clocks and to find the propagation time τ between them. Let them do the following; where t' and t denote the time according to the clocks at A and B, respectively.

GIVEN:

Assume A is ε seconds behind B

i.e., when $t' = 0$, $t = \varepsilon > 0$,

or $t' = t - \varepsilon$

A and B have identical cryptographic key-stream generators x( )

B has a programmable matched filter $h_B(t)$

with an impulse response set to:

$$h_B(t) = \begin{cases} x(T-t), & 0 < t < N\delta \\ 0, & \text{elsewhere} \end{cases}$$

WHERE:

T = time adjustment $> \varepsilon + \tau + N\delta$
(allows it to await event to come)

N = Number of chips in sequence to be correlated

δ = Chip duration in seconds

THEN:

A transmits $x(t') = x(t - \varepsilon)$

B receives τ seconds later

THEREFORE:

$$r(t) = x(t - \varepsilon - \tau) * h(t)$$

$$= \int_{-\infty}^{\infty} x(\alpha - \varepsilon - \tau)\, h(t - \alpha)\, d\alpha$$

$$= \int_{-\infty}^{\infty} x(\alpha - \varepsilon - \tau)\, x(T-t + \alpha)\, d\alpha$$

Which is maximum for

$$-\varepsilon - \tau = T-t$$

i.e. $$t = T + \varepsilon + \tau$$

BUT:

B knows T (offset introduced to await transmission)

THEREFORE:

B can calculate $\varepsilon + \tau$

SIMILARLY:

Assume A has a programmable matched filter: $h_A(t)$ with impulse response set to:

$$h_A(t) = \begin{cases} x(T-t'), & 0 < t' < N\delta \\ 0, & \end{cases}$$

(Although his time origin is wrong, A's offset forward by T from the origin is not wrong.)

NOW:

Now in order to make it easy for A to determine , B transmits a signal advanced by $\varepsilon + \tau$

A receives τ seconds later

THUS:

$$r(t) = x(t + \varepsilon) * h(t)$$

$$= \int_{-\infty}^{\infty} x(\alpha + \varepsilon)\ h(t - \alpha)\ d\alpha$$

$$= \int_{-\infty}^{\infty} x(\alpha + \varepsilon)\ x(T-t + \alpha)\ d\alpha$$

which is maximum for $\varepsilon = T-t$

i.e., $t = T - \varepsilon$

BUT:

A knows T

THERFORE:

A knows ε and can adjust his t' to be synchronized to t, to within an accuracy of δ seconds.

4. Performance

If only one PMC initiates a level one message then the number of levels required to synchronize the network is equal to the "diameter" of the connected graph formed by the existing connectivities within the network. Simply, the diameter of a graph is its maximum minimum (longest shortest) path between any two nodes in the network. This is sometimes known as the longest geodesic of a graph. (HAR 72)

The amount of time required therefore is D x(Δ)t, where D is the network diameter and (Δ)t is the interval between successive level transmission.

B. DISTRIBUTED NETWORK MANAGEMENT

1. Network Types

Two basic types of networks are available for the transmission of digital data. The line or circuit-switched type and the message or packet-switched type. These are distinct techniques for communicating among the nodes of the network and combinations of these switching strategies are possible as in "pacuit" switching. (GER 79)

Line-switched networks connect the source of the transmission and its destination (sink) by a communications path that is established at the beginning of the connection, and cancelled when the desired transfer of data is completed or when disrupted by a failure. In different routing strategies, described below, paths may remain fixed or be changed (not cancelled) during the existence of the connection. Such a subset of line-switching is called "virtual" line-switching. Data is forwarded according to designated paths but messages corresponding to different connections are multiplexed together on each link so that the portion of the link capacity used by each message is varied according to its transmission requirements.

Message-switched networks may utilize different paths from source to sink for almost every message generated. Such paths are not predetermined but are incrementally determined by intervening relay nodes. The selection of these neighbors is the heart of the routing strategy. Packet-switching is a fundamental subset of message switching in which messages are subdivided into a number of smaller

segments each of which is labelled by the network as a message destined for the same sink. (SID 79)

2. Control of Networks

Two main approaches exist in the control of routing procedures within a network. There is the centralized approach, of which PLRS is a good example, and the decentralized or distributed approach of which presently ARPANET is an operating example.

Little more needs to be said about centralized adaptive techniques beyond the observations noted in Chapter III. While achieving such goals as "optimal" routing is made easier, the centralized adaptive procedures are inherently weakened by the imposition of requirements for communicating enormous amounts of status information back to the central node. In addition, the central memory (MU) could become separated from a portion of its subscriber network or vice versa due to link failures. Such a situation is highly undesirable in a tactical C3 system. Further the requirements for all routing to be centrally processed creates unbalanced demands on network link bandwidth which could limit the actual size of the supported network. The addition of sub-controllers (NCU) can help to alleviate some of the weakness that centralized systems display and the PLRS/JTIDS HYBRID uses such an approach.

The distributed adaptive control schemes have none of the inherent inefficiency or unreliability of fixed routing nor the unreliability and size limits of centralized

control measures. Each node performs the necessary computations to make routing decisions in conjunction with its adjacent nodes (neighbors). This is usually manifested by a table of routing information being maintained at each node to identify the output links to be used from it for each destination in the network. Such tables could be updated periodically or asynchronously (as needed) by using the routing information each node collects internally and that which it receives from its neighbors. In an N node network each table can have up to N-1 entries. Each is a measure of the estimated minimum distance from that node to every other node in the network as well as the neighbor to which a message is next relayed.

Distributed routing systems are not all strengths with no weaknesses. Without any global knowledge of nodal status within the network loop freedom is difficult to guarantee. Failsafe operation of the network management procedure becomes much more difficult to achieve. These are some of the challenges of this fascinating subject. (SID 79)

3. Routing Strategies

Routing strategies can be measured by the yardstick of their dynamicism. Typically we have static, quasi-static and dynamic strategies.

Purely static or deterministic routing sets up the rules which determine fractional traffic loads on each link prior to network operation. Further they determine actual path assignments from each node to each other node.

To do this they use knowledge about node location, connectivities, and link capacities as well as overall "loading" and requirements of the network. Fixed routing implies solutions to questions like how a message goes from A to B which are neither adaptive to changes in the network status or requirements, nor reliable enough for much practical use. Their simplicity, however, makes them appealing in the network design phase.

On the other end of the spectrum are completely dynamic routing strategies which allow nearly continuous change of routes as a function of time and network loading conditions (traffic input, queue lengths, link and nodal failures and additions). While responsiveness to network requirements is desirable the large amount of "overhead" per message through the network is an undesirable side effect. This overhead information is required in order to restructure addressing at each node in response to the changes in the network.

Quasi-static routing is adaptive in nature but forbids continuous generation of new routing solutions. Here routing assignments may be modified only periodically or when a network need arises due to an extreme situation. Link and node failures or additions are typical events which would precipitate such action. In order to allow adaptivity the quasi-static routing procedure must be able to sense and react to changes in the network topology and loading conditions. Adaptivity to failures is very important in both our scenario and analytically in order to maintain a good grade of service to the network subscribers. (SID 79)

VI. A QUASI-STATIC DISTRIBUTED ROUTING PROTOCOL

A. SYNOPSIS

1. Background

The second question to be investigated is the network management and routing of information in a distributed network. Without this capability synchronous operations previously guaranteed are of little advantage over present and planned systems. Implicit in the "distributed" operations of our scenario are requirements that such operations include the properties of loop freedom for each destination at all times, adaptivity to changes in network flow and topology, and completely failsafe operation. Failsafe as used here means that given an arbitrary number and location of failures and/or additions in the network structure, the network recovers in finite time after the last change to provide routing paths between all remaining physically connected nodes. These properties are obtained using asynchronous computation of distributed status information in lieu of any one source having global topological knowledge. The algorithm investigated here is due to Segall and Merlin. (SEG 79)

2. Procedures

In the discussion that follows, the simplifying assumption is made that all traffic is destined to a single (sink) node. In practice, the procedure to be described would be iterated once for each possible sink in turn.

Each node knows only who its next node (preferred neighbor) on the route to a given destination is. Each node is responsible for choosing new preferred neighbors (PI's) based upon updating its own routing tables. Such updates are coordinated by the protocol via control messages sent between adjacent nodes. It has been shown in (SEG 79) that for a given destination the set of routes maintained by the protocol are loop-free at all times, and that whenever no failures occur they form a spanning tree rooted at the sink (i.e., they connect all nodes). To each link in the network a strictly positive "distance" or weight is assigned which represents the cost of using that link. Such costs will vary with time according to link utilization and other factors such as queue length. The length of a path then is the sum of the distances of its links.

Destinations (sinks) may asynchronously trigger the protocol to start new "update cycles" to generate new routing solutions based on new topology or loading conditions. Such adaptivity is a function of user specified goals, e.g., maximum throughput, minimum average delay and so forth. A cycle first propagates uptree while modifying the distance estimates from nodes to the destination and then comes back downtree while updating PI's. Each cycle tends to find routes with short paths from each node to the destination. Assuming time-invariant link weights it finds the strict minimum within a finite number of iterations.

When a link fails, notice of its failure is sent

down to the sink of the tree and up to disconnected nodes above it by the nodes adjacent to the failure. When such notice arrives at the sink it triggers a new update cycle and the protocol guarantees that within finite time all nodes physically connected, even if by presently unused links, to the sink will have a loop-free route to it. This property holds for multiple topological changes, and even if such changes occur while the protocol is active and an update is in progress. The recoverability of the protocol is achieved without employing a "time-out" in its operation. Such capability enhances not only the analysis and structured implementation of the protocol but gives it wide ranging applicability to tactical C3 problems.

The protocol is intended for use in quasi-static routing of data in communications networks. Application of it to a line-switched network would imply that the routes generated would be used to assign paths to a new or disrupted call. The link weights might represent delays. Thus in steady state the minimum delay route for the new call is found. Should the links represent incremental delays then the path minimizes the average network delay. In message-switched systems where the PI is the first "hop" of the present best estimated route to the sink, increasing the fraction of messages sent over the shortest path seems entirely practical and the preferred routing method for our scenario.

One major benefit of this protocol is that it can replace the saturation or "flooding" used in some networks to locate mobile subscribers and to select the routing paths. This concept has the advantages of saturation routing without requiring time out while providing a route selected not only on the basis of instantaneous congestion but on averaged quantities. (SEG 79)

3. Nodal Functions

A detailed listing of the algorithm for an arbitrary node is provided in Appendix B. The actual number of different operations performed by any node is small. A node receives and sends messages of four types. It updates its routing tables based on information contained in those messages.

For example whenever a node i receives a message type MSG (see Table 1 Appendix B) from neighbor 1, the receiving node estimates and stores its distance through 1 to the SINK. The required data to perform this updating is contained in the MSG message from 1. As another example when i has received MSG transmissions from all his neighbors i transmits an MSG to its PI and then determines a new PI if one exists based on the new information in its tables. This is done by choosing the new PI as the neighbor which provides minimum estimated distance from i to SINK. (SEG 79)

B. SIMULATION

1. Method

> "A theory has only the alternative of being right or wrong. A model has a third possibility; it may be right but irrelevant."
>
> Manfred Eigen

The simulation conducted of the protocol discussed above took the form of an examination of the incremental steps the algorithm performed in a volatile operating network. Each facet (capability) of the protocol was demonstrated, not by way of proving any separate characteristic, but rather to examine the data storage requirements necessary to support execution of the algorithm at any node in a network. Rigorous analytical proofs of all stated properties of the algorithm are included in (SID 79). In order to display the dynamic capability of the protocol in a static display, like a thesis, a method of data presentation was developed to examine the data status and message processing and handling capabilities of the algorithm. A computer simulation of it was developed in SIMSCRIPT II.5 (Appendix A). This was intended to be a completely general representation of the statement of the functioning algorithm whose operation was summarized by the authors as a finite state machine (FSM) in which transitions between states were triggered by the arrival of control messages from neighbors and actions taken as a result of those transitions.

2. Results

In answer to the question of data storage requirements in support of the distributed network management

algorithm, both analytical techniques and simulation results were used. By examining the data stored by the algorithm at each node and noting the nature of each data set it was quickly determined that of the ten data elements six varied linearly with the number of nodes (N) in the network. Four were found to be directly proportional to both the number of nodes and the number of links (L) connecting them. Thus ((6N) + (4NL)) data elements need to be maintained by each node in support of the algorithm operation. This also proves to be the number of data elements stored by each node in the simulation. See Table 1 of Appendix C for a list of the variables of the algorithm. While each data element is in general less than five characters in length the total characters stored in a node may become excessive for a microprocessor application. This would be the case in a very large network, more than fifty nodes for example. In such a network with one hundred links the nodal storage requirements for protocol related data alone is over 11,000 characters. Superimposed upon this would be application programs and other local protocols necessary for network operations and use by subscriber as well as message buffers for communications functions plus ROM coding of protocol instructions.

These results begin to indicate that some sub-network structure might be beneficial to the storage requirements of individual nodes as well as to network responsiveness. By limiting the bulk memory requirements their size can be

kept to a minimum for ease of transportation in a highly mobile environment. They must still be capable of full support of their subscribers through the use of distributed functions. So a trade off may need to be made in their size and weight versus the desire for completely distributed operations.

The ability of the algorithm to function in a dynamic network was not doubted at the outset. The proofs which are appended to the source document were clear and thorough leaving little doubt in the readers mind that such a system could be implemented. The purpose of the simulation therefore was not by way of a proof but rather a demonstration. It is intended for operational and design personnel to make them aware that such a powerful algorithm exists and can possibly be tailored to the needs of tactical C3 systems of the future. The various capabilities of the algorithm to handle network failures and additions in conjunction with routine network management functions were demonstrated.

The data which follows was generated from the simulation listing of Appendix A. The initial status of all links shown were operational and the weights are as indicated in the network diagrams. The changes in link weights are specified in the narrative explanation of the data on each page. The final tree after simulation is also provided for information.

The fact that all simulation runs converged to short path solutions in one iteration should not be misconstrued to mean that all such changes in network status can be solved in one pass. Rather in these simple examples the singularity of failures/changes without compunding ones allowed the algorithm to operate in its most efficient manner without need for second or subsequent trys to find the optimum solution as would probably be necessary in more complex and volatile networks.

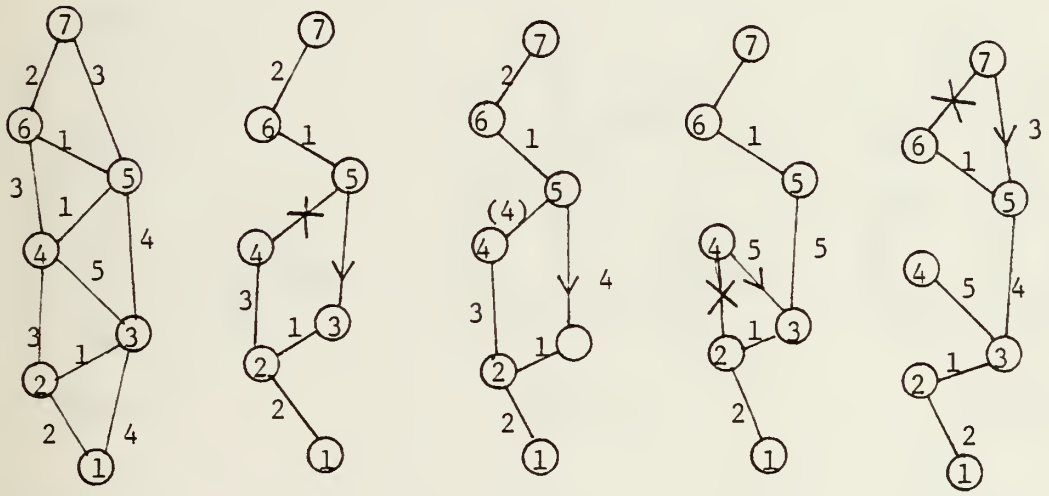
NETWORK - 1

N = 7
L = 11

1. Number of transmissions required to complete update cycle (root to root) 22

2. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure in (4,5) 1 / 24
   B. Link weight change in (4,5) from 1 to 4 1 / 24

3. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure near root (2,4) 1 / 28
   B. Link failure in a branch (6,7) 1 / 24



NETWORK TREE 2A TREE 2B TREE 3A TREE 3B

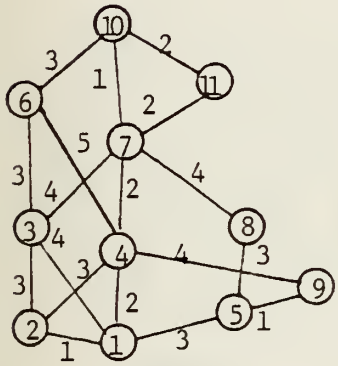
NETWORK - 2

N = 11
L = 17

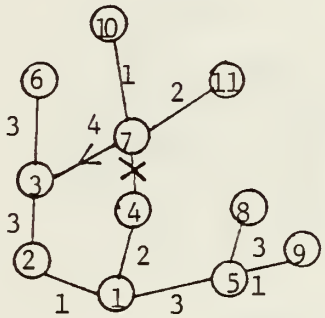
1. Number of transmissions required to complete an update cycle (root to root) 34

2. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure in (4,7) 1 / 43
   B. Link weight change in (5,8) from 3 to 6 1 / 36

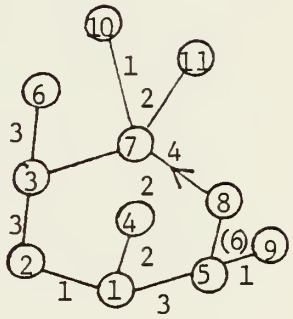
3. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure near root (1,4) 1 / 52
   B. Link failure in branch (7,11) 1 / 34



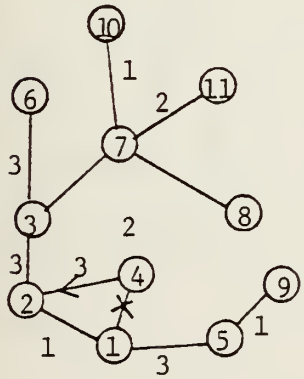
NETWORK



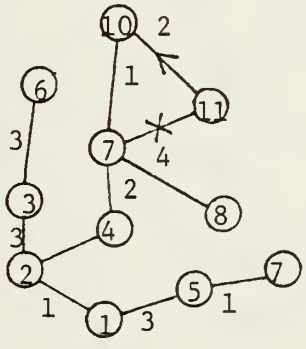
TREE 2A



TREE 2B



TREE 3A



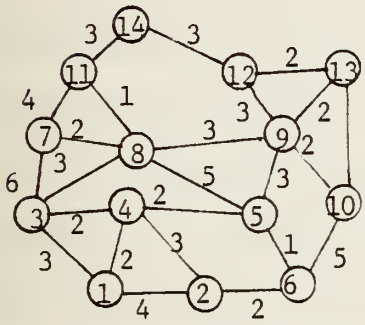
'TREE 3B

N = 14
L = 24

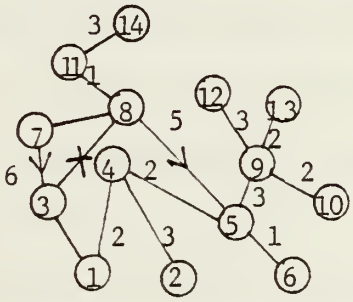
1. Number of transmissions required to complete an update cycle (root to root) 48

2. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure in (3,8) 1 / 62
   B. Link weight change in (4,5) from 2 to 6 1 / 50

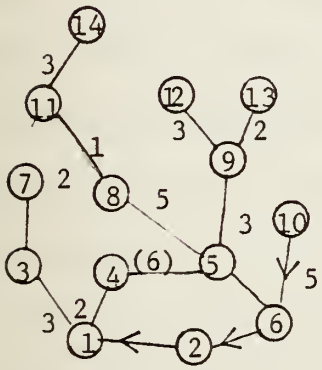
3. Number of iterations/transmissions to find a new short path solution given:
   A. Link failure near root (1,3) 1 / 66
   B. Link failure in branch (11, 14) 1 / 48



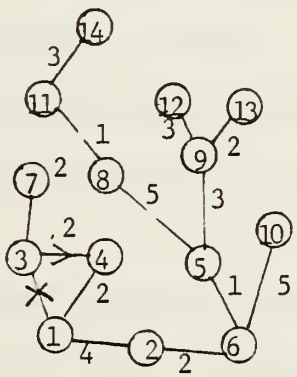
NETWORK



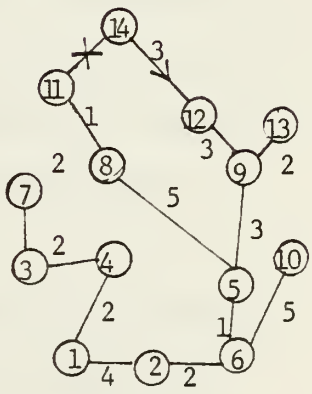
TREE 2A



TREE 2B



TREE 3A



TREE 3B

VII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The development of a distributed C3 system as a post MTACCS concept appears both desirable and possible. In the face of the Radio Electronic Combat (REC) threat few detractors can say that such system capabilities are not desirable. No commander would say that he did not want such flexible support for his tactical decisions in a mobile and destructive conflict.

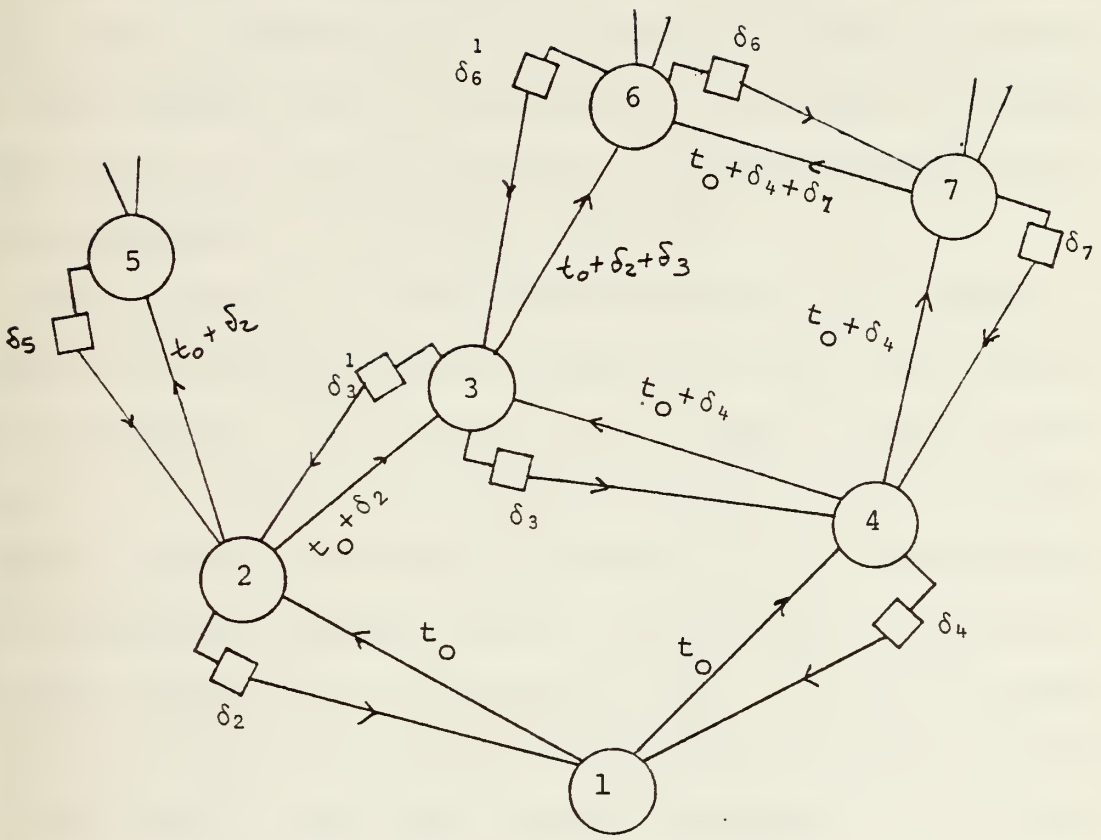
It is possible because it has been demonstrated that the two greatest initial problems which blocked distributed system implementation in a tactical environment have been removed. The key concept is the ability to manage the network in a loop-free and failsafe manner. Secondary to this is the ability to guarantee network synchronization given temporary catastrophic failure in synchronous network operations. Hierarchical C3 systems may be an architecture whose time is passing, and it remains only for operational and design personnel to sufficiently delineate the desired capabilities of a system like MCC to begin its actual development.

The problem of desired sub-network structure within a distributed system appears to be a series of trade-offs between the amount of structure versus the amount of distribution. Each facet brings with it those strengths and

weaknesses previously discussed. It is desired to retain as many of the advantages of distributed operation as possible while eliminating the unmanageably large delay times required for new solutions to be generated in large networks. Further it is desirable not to build in so much structure that the problems associated with present architectures resurface. Whether such trade-offs would be a function of the location of nodes, connectivity, or other factors either singly or in combination is unknown. A recent paper on shortest path algorithms in communications (YEN 79) networks has suggested a framework for a possible solution to this problem. Its description follows, but like the MCC concept presented previously, this proposal is a subject for further investigation.

This algorithm operates under the following assumptions. Each pair of nodes is connected by at least one pair of bi-directional links. The delays on these links are different for different directions of travel (queue lengths are assumed to constitute the major delay). All timing and service messages are jumped to the head of the message queue to ensure proper timing on outbound links.

At time T(0) the sink transmits to its neighbors who hear and record the message. They in turn transmit to their neighbors at t(0) + δ(X), where δ(X) = local queue delay time for the link to the sink. Each of their neighbors upon hearing this transmission, waits until T(0) + δ(X) + δ(Y), where δ(Y) = their local queue delay toward the sink, then they transmit as before. (See Figure 7.)



δ DELAY

NODE

QUEUE

NOTE:

let

$\delta_2 < \delta_4$

$\delta_3 < \delta_3^1$

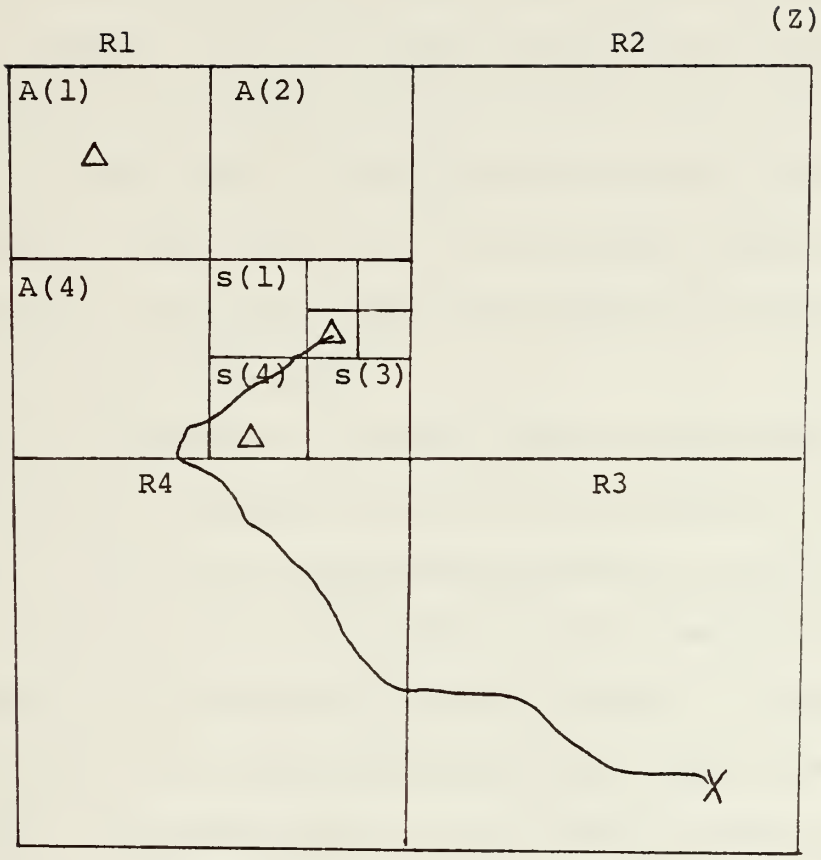
FIGURE 7. A Shortest Path Algorithm

Should a node be the recipient of two or more transmissions from its neighbors it will of course forward only the earliest received as this will ensure those above it in the network of the shortest path to the sink.

Since each node waits until time t(0) + (all cumulative δ's) before transmitting, it is assured of never receiving a later message with a shorter path to the sink. Further, each node will only be required to transmit once, and no acknowledgement is required. (YEN 79)

The proposal for sub-network structure is a direct result of the previously discussed work (See also Figure 8 which follows). Let the set of all nodes be known as the set (Z). Let this be divided into regions (R(i)) and let these be further divided into areas (A(j)). Let each area be divided into sectors (s(k)), and let each of these be further divided into sub-sectors (Q(1)). Let the divisor be four. With four nodes in each Q(1) the network capacity is 1024 nodes. Thus this example translates to a notional two Marine Amphibious Brigade (MAB) size network. (This example assumes a PLRS notional MAB of 400 nodes).

Since each station only transmits once to achieve local synchronization let those stations (nodes) in R(1).A(3). S(2).Q(4) (See Figure 8) do this to achieve this state. Let one of these nodes be designated, by an as yet undetermined method, as the Primary Station (PS). Let there be one PS for each level of division, e.g., there is one PS acting for R(1) in A(1) and there is one PS acting for A(3) in S(4)



(Z) = set of all nodes
R(l) = Region
A(j) = Area
S(k) = Sector
Q(l) = Sub-sector

△ = Primary station

FIGURE 8. Distributed Network Sub-structure

and for S(2) in Q(4). Thus there is a PS acting for each level of division but resident one level below that level. Let each station in each level quadrant find shortest path solutions to each PS in the next level higher quadrant, Q to S for example. Then let each station find the shortest path to each PS two levels above itself, Q to A for example. Finally let each node find the shortest path to each PS 3 levels above itself, Q to R. The algorithm indicates that this procedure take four times the logarithm (base 4) of N transmissions by each node.

Then as indicated in Figure 8 when an arbitrary node X wanted to send a message to R(1).A(3).S(2).Q(4) it would proceed by finding the shortest path out of its Region. Then the message would be routed in R(4) towards R(1) via the shortest path to the PS in A(1) of R(1). Finally upon crossing into R(1) the local nodes there would have knowledge of shortest paths to A(3) and so divert the message from the PS in A(1) toward A(3) and upon crossing that boundary similarly to S(2) and finally to Q(4).

While it is recognized that this algorithm and its presentation in this example does not represent the strictest short path solution its virtues are simplicity and low number of transmission as well as small storage requirements for routing information required to implement it while still achieving a short path in such a network.

B. RECOMMENDATIONS

It is recommended that the Marine Corps closely monitor the present test of ARPANET technology at Fort Bragg and maintain close liaison with the Army in the development and implementation of the ADDS packet-radio test to follow. It is further recommended that the Marine Corps begin to examine in greater detail the concept of sub-network structure in a distributed C3 system with a view towards optimizing the trade-off in node size and capability. As a possible source for such investigation it is also recommended that the officer-graduate students of the Marine Corps be utilized as a resource to accomplish this and other related tasks in the future development of the post MTACCS C3 systems. This group of individuals possess the time and resources to provide valuable input to such decisions and their efficient use by the Marine Corps could help to limit the initial cost of the development of such systems in the future.

APPENDIX A

DISTRIBUTED NETWORK MANAGEMENT SIMULATION LISTING

```
//  EXEC SIM25CLG
//SIM.SYSIN DD *
''THIS IS THE BASIC SIMULATION OF MY THESIS''
''IT IS DESIGNED TO SIMULATE THE DISTRIBUTED NETWORK ''
''MANAGEMENT ALGORITHM OF ADRIAN SEGALL AT ITS MOST BASIC''
''LEVEL FOR ONE TREE (EVENTUALLY FOR MULTIPLE TREES)''
''THE ENTIRE SIMULATION WILL SHOW THE CAPABILITY OF THE''
''ALGORITHM TO HANDLE LINK FAILURES, NETWORK ADDITIONS ''
''AND ROUTINE NETWORK MANAGEMENT. IT IS DESIRED TO BE ABLE''
''TO DETERMINE HOW MUCH STORAGE OF NETWORK MANAGEMENT DATA''
''IS REQUIRED BY EACH NODE/ AND HOW POWERFUL A PROCESSOR''
''THIS ALGORITHM  MIGHT REQUIRE IN A MAN-PACK REALIZATION.''
''OTHER CONSIDERATIONS ARE WHAT IMPACT SUCH A CAPABILITY, ''
'' I.E. DISTRIBUTED NETWORK MANAGEMENT; EXTRAPOLATED''
''TO DISTRIBUTED COMMUNICATIONS SYSTEMS,MIGHT HAVE ON THE ''
''COMMAND AND CONTROL STRUCTURE OF THE MARINE CORPS IN THE''
''1990'S WITH SPECIFIC ATTENTION TO THE COMMUNICATIONS ''
''NETWORK IMPACT.''


PREAMBLE
NORMALLY MODE IS INTEGER
GENERATE LIST ROUTINES
PERMANENT ENTITIES
   EVERY NODE HAS A PI,A DI, A MAXI,AN NI,A CT, A NBR,
     A STATE, AND OWNS THE QUE
  EVERY SINK HAS A  SMAXI,A SCT,
    A SSTATE, AND OWNS THE QUEUE
TEMPORARY ENTITIES
   EVERY MESSAGE HAS A SNK,AN L,AN M,A D,AN I,A TYPE,
     AN ORIGIN, A DEST AND MAY BELONG TO THE QUE AND
     THE QUEUE
DEFINE FIL AND SFIL AS 2-DIMENSIONAL,ALPHA ARRAYS
DEFINE N,K,MM AS  INTEGER VARIABLES
DEFINE ZIL,DILL,DIL,NIL,SDIL,SNIL AS 2-DIMENSIONAL,INTEGER
ARRAYS
DEFINE STATE,SSTATE, AND TYPE AS ALPHA VARIABLES
DEFINE CONNECTIVITY AND TREE AS 2-DIMENSIONAL,INTEGER ARRAYS
EXTERNAL EVENTS ARE START.UP,NODE.ADD,LINK.FAIL
EVENT NOTICES INCLUDE STOP.SIMULATION, OUT.PUT
EVERY RECEIVE.A.MESSAGE HAS A TEXT AND A HOME
END
MAIN
READ N ''THE DIMENSIONS OF THE CONNECTIVITY ARRAY''
RESERVE CONNECTIVITY(*,*) AS N BY N
READ CONNECTIVITY ''READS IT A ROW AT A TIME FROM CARDS''
RESERVE DIL,ZIL,DILL,NIL,SDIL, AND SNIL AS N BY N
READ DIL
RESERVE FIL AND SFIL AS N BY N
READ FIL  ''UP/DOWN LINK STATUS''
READ MM ''THE DIMENSIONS OF THE DIRECTED TREE, 1 TO START''
RESERVE TREE AS MM BY MM
READ TREE ''READS IT A ROW AT A TIME FROM CARDS''
READ K ''SIZE OF NETWORK''
CREATE EVERY SINK(K) ''SIZES NODE ATTRIBUTES''
CREATE EVERY NODE(K) ''-DO-''
FOR EACH NODE LET STATE(NODE)="S1"
FOR EVERY NODE READ PI(NODE),DI(NODE)
FOR EVERY NODE READ MAXI(NODE)
SCHEDULE A OUT.PUT AT TIME.V + 39.9
SCHEDULE A STOP.SIMULATION AT TIME.V + 40.0  ''LONGEST +''
START SIMULATION ''GOES TO FIRST EVENT, CARD AFTER DATA ''
END
EVENT START.UP
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EVENT START.UP ENTERED, THE TIME IS ***.**
READ RVCR
CREATE A MESSAGE
  LET TYPE(MESSAGE)="REQ"
READ SNK(MESSAGE),DEST(MESSAGE),D(MESSAGE),M(MESSAGE),
   ORIGIN(MESSAGE)
```

```
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND RVCR AT
   TIME.V + 1.C
LIST ATTRIBUTES OF MESSAGE
RETURN
END
EVENT NODE.ADD
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EVENT NODE.ADD ENTERED, THE TIME IS ***.**
READ RVCR
CREATE A MESSAGE
   LET TYPE(MESSAGE)="WAKE"
READ SNK(MESSAGE),DEST(MESSAGE),D(MESSAGE),M(MESSAGE),
   ORIGIN(MESSAGE)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 2.2
RETURN
END
EVENT LINK.FAIL
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EVENT LINK.FAIL ENTERED, THE TIME IS ***.**
READ RVCR
CREATE A MESSAGE ''MESSAGE DOWN-TREE TO SINK''
   LET TYPE(MESSAGE)="FAIL"
READ SNK(MESSAGE),DEST(MESSAGE),D(MESSAGE),M(MESSAGE),
   ORIGIN(MESSAGE)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 2.3
READ RVCR
CREATE A MESSAGE ''MESSAGE UP-TREE TO DISCONNECTED NODES''
   LET TYPE(MESSAGE)="FAIL"
READ SNK(MESSAGE),DEST(MESSAGE),D(MESSAGE),M(MESSAGE),
   ORIGIN(MESSAGE)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 2.31
RETURN
END
EVENT RECEIVE.A.MESSAGE GIVEN COPY AND CATCHER
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     EVENT RECEIVE.A.MESSAGE ENTERED, THE TIME IS ***.**
'BASIC'
LET NODE=CATCHER
IF TYPE(COPY) ="MSG" AND STATE(NODE) ="S1",
   LET NIL(DEST(COPY),ORIGIN(COPY))=M(COPY)
  LET DIL(NODE,ORIGIN(COPY))=D(COPY)+DIL(NODE,ORIGIN(COPY))
   IF FIL(NODE,ORIGIN(COPY))="READY"
   LET FIL(NODE,ORIGIN(COPY))="UP"
   ALWAYS
   IF ORIGIN(COPY)=PI(NODE),
     LET CT(NODE)=0
     LET STATE(NODE)="S2"
   GO TO BFSM12
    ELSE
    RETURN
ELSE
IF TYPE(COPY)="MSG" AND DEST(COPY)=SNK(COPY) AND
SFIL(NODE,ORIGIN(COPY))="HRD" AND
STATE(NODE)="S2"
LET STATE(NODE)="S1"
PRINT 1 LINE AS FOLLOWS
     PROPER COMPLETION
RETURN
ELSE
IF TYPE(COPY) ="MSG" AND STATE(NODE) ="S2",
   LET NIL(DEST(COPY),ORIGIN(COPY))=M(COPY)
   LET DIL(NODE,ORIGIN(COPY))=D (COPY) +
   DIL(NODE,(ORIGIN(COPY)))
FOR II=1 TO N, WITH CONNECTIVITY(NODE,II) GT 0
   AND NIL(NODE,II) LE 0,
FIND THE FIRST CASE
IF NONE, GO TO BFSM21
   ELSE
   RETURN
```

```
ELSE
IF TYPE(COPY)="FAIL" AND STATE(NODE)="S1" AND
   ORIGIN(COPY)=PI(NODE)
   LET FIL(NODE,ORIGIN(COPY))="DOWN"
   LET STATE(NODE)="S3"
   LET CT(NODE)=0
   GO TO BFSM12
ELSE
IF TYPE(COPY)="FAIL" AND STATE(NODE)="S2" AND
   ORIGIN(COPY)=PI(NODE)
   LET FIL(NODE,ORIGIN(COPY))="DOWN"
   LET STATE(NODE)="S3"
   LET CT(NODE)=0
   GO TO BFSM23
ELSE
IF TYPE(COPY)="WAKE"  AND PI(NODE)=0 AND DI(NODE)=99999
   FOR KS=1 TO N, WITH FIL(NODE,KS)="UP" AND
   NIL(NODE,KS)=MAXI(NODE)
   COMPUTE MINDIL AS THE MINIMUM OF DIL(NODE,KS)
   LET PI(NODE)=KS
   LET NI(NODE)=MAXI(NODE)
   LET DI(NODE)=MINDIL
   GO TO BFSM32 ''REATTACHING NODES ONLY''
ELSE
IF TYPE(COPY)="WAKE" AND PI(NODE) NE 0 ''UPDATE TABLES''
   LET CONNECTIVITY(NODE,ORIGIN(MESSAGE))=D(MESSAGE)
   LET CONNECTIVITY(ORIGIN(MESSAGE),NODE)=D(MESSAGE)
RETURN
ELSE
IF TYPE(COPY)="REQ" AND DEST(COPY) = SNK(COPY)
FOR KZ=1 TO N, WITH CONNECTIVITY(NODE,KZ) GT 0
DO
   CREATE A MESSAGE ''START NEW UPDATE CYCLE''
      LET TYPE(MESSAGE) = "MSG"
      LET M(MESSAGE) = MAXI(NODE) + 1
      LET SNK(MESSAGE)= SNK(COPY)
      LET ORIGIN(MESSAGE)=DEST(COPY)
      LET DEST(MESSAGE)=KZ
      LET SFIL(NODE,ORIGIN(COPY))="HRD"
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LIST ATTRIBUTES OF MESSAGE
LET MAXI(NODE)=M(MESSAGE)
LOOP
DESTROY MESSAGE CALLED COPY
LET STATE(NODE)="S2"
RETURN
ELSE
IF TYPE(COPY)="REQ" AND PI(NODE) NE 0
   CREATE A MESSAGE
      LET TYPE(MESSAGE)="REQ"
      LET SNK(MESSAGE)=SNK(COPY)
      LET ORIGIN(MESSAGE)=DEST(COPY)
      LET M(MESSAGE)=M(COPY)
      LET DEST(MESSAGE)=PI(DEST(COPY))
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LIST ATTRIBUTES OF MESSAGE
DESTROY MESSAGE CALLED COPY
RETURN
ELSE
PRINT 1 LINE AS FOLLOWS
      ERROR ---INCORRECT MESSAGE TYPE RECEIVED---
RETURN
'BFSM12'
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     BFSM12 ENTERED, THE TIME IS ***.**
FOR IJ= 1 TO N,
WITH NIL(NODE,IJ) EQ M(COPY) AND FIL(NODE,IJ)="UP" AND
DILL(NODE,IJ) GT 0
COMPUTE MINDIL AS THE MINIMUM OF DILL(NODE,IJ)
PRINT 1 LINE WITH MINDIL AS FOLLOWS
```

```
MINDIL = ***
IF M(COPY) GT MAXI(NODE),
   LET MAXI(NODE)=M(COPY)
ALWAYS
FOR KK=1 TO N,
   WITH CONNECTIVITY(NODE,KK) GT 0 AND
   KK NE PI(NODE) AND FIL(NODE,KK)="UP"
DO
 CREATE A MESSAGE
   LET DEST(MESSAGE)= KK
   LET TYPE(MESSAGE)= "MSG"
   LET SNK(MESSAGE)= SNK(COPY)
   LET D(MESSAGE)= DI(NODE)
   LET M(MESSAGE)=MAXI(NODE)
   LET ORIGIN(MESSAGE)= DEST(COPY)
LIST ATTRIBUTES OF MESSAGE
   SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LOOP
FOR IX=1 TO N, WITH CONNECTIVITY(NODE,IX) GT 0
   AND NIL(NODE,IX) LE 0, FIND THE FIRST CASE
   IF NONE, GO TO BFSM21
ALWAYS
DESTROY MESSAGE CALLED COPY
RETURN
'BFSM21'
PRINT 1 LINE WITH TIME.V AS FOLLOWS
   BFSM21 ENTERED, THE TIME IS ***.**
FOR JK=1 TO N, WITH DILL(NODE,JK) GT 0
COMPUTE NEWPI AS THE MINIMUM (JK) OF DILL(NODE,JK)
LET PI(NODE)=NEWPI
FOR KJ=1 TO N, WITH FIL(NODE,KJ)="UP"
LET NIL(NODE,KJ)=0
LET CT(NODE)=1
LET STATE(NODE)="S1"
CREATE A MESSAGE
LET TYPE(MESSAGE)="MSG"
LET SNK(MESSAGE)=SNK(COPY)
LET D(MESSAGE)=DI(NODE)
LET ORIGIN(MESSAGE)=DEST(COPY)
LET M(MESSAGE)=MAXI(NODE)
LET DEST(MESSAGE)=PI(CATCHER)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LIST ATTRIBUTES OF MESSAGE
DESTROY MESSAGE CALLED COPY
RETURN
'BFSM13'
PRINT 1 LINE WITH TIME.V AS FOLLOWS
     BFSM13 ENTERED THE TIME IS ***.**
   LET DI(NODE)=99999 ''APPROX INFINITY''
   LET NIL(NODE,ORIGIN(COPY))=M(COPY)
FOR IP=1 TO N, WITH FIL(NODE,IP)="READY"
   IF NIL(NODE,IP) GT ZIL(NODE,IP)
   LET FIL(NODE,IP)="UP"
   LET NIL(NODE,IP)=0
ALWAYS
FOR IQ=1 TO N, WITH CONNECTIVITY(NODE,IQ) GT 0
   AND IQ NE PI(NODE) AND FIL(NODE,IQ)="UP"
DO
CREATE A MESSAGE
   LET D(MESSAGE)=DI(NODE)
   LET M(MESSAGE)=MAXI(NODE)
   LET ORIGIN(MESSAGE)= NODE
   LET DEST(MESSAGE)= IQ
   LET SNK(MESSAGE)=SNK(COPY)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND IQ AT
   TIME.V + 1.0
LOOP
DESTROY MESSAGE CALLED COPY
LET PI(NODE)=0
LET CT(NODE)=1
```

```
RETURN
'BFSM22'
PRINT 1 LINE WITH TIME.V AS FOLLOWS
      BFSM22 ENTERED THE TIME IS ***.**
   LET DI(NODE)=99999 ''APROX INFINITY''
   LET NIL(NODE,ORIGIN(COPY))=M(COPY)
FOR IR=1 TO N, WITH FIL(NODE,IR)="READY"
   IF NIL(NODE,IR) GT ZIL(NODE,IR)
   LET FIL(NODE,IR)="UP"
   LET NIL(NODE,IR)=0
ALWAYS
FOR KQ=1 TO N, WITH CONNECTIVITY(NODE,KQ) GT 0
   AND KQ NE PI(NODE) AND FIL(NODE,KQ)="UP"
DO
CREATE A MESSAGE
   LET D(MESSAGE)=DI(NODE)
   LET M(MESSAGE)=MAXI(NODE)
   LET ORIGIN(MESSAGE)= NODE
   LET DEST(MESSAGE)=KQ
   LET SNK(MESSAGE)=SNK(COPY)
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LOOP
DESTROY MESSAGE CALLED COPY
LET PI(NODE)=0
LET CT(NODE)=1
RETURN
'BFSM32'
PRINT 1 LINE WITH TIME.V AS FOLLOWS
      BFSM32 ENTERED, THE TIME IS ***.**
FOR KR=1 TO N, WITH CONNECTIVITY(NODE,KR) GT 0
   AND KR NE PI(NODE) AND FIL(NODE,KR)="UP"
DO
CREATE A MESSAGE
   LET D(MESSAGE)=DI(NODE)
   LET M(MESSAGE)=MI(NODE)
   LET ORIGIN(MESSAGE)=NODE
   LET DEST(MESSAGE)=KR
SCHEDULE A RECEIVE.A.MESSAGE GIVEN MESSAGE AND DEST AT
   TIME.V + 1.0
LOOP
DESTROY MESSAGE CALLED COPY
   LET CT(NODE)=1
FOR KT=1 TO N, WITH FIL(NODE,KT)="READY" AND NI(KT) GT
 ZIL(NODE,KT)
   LET FIL(NODE,KT)="UP"
   LET NIL(NODE,KT)=0
RETURN
END
EVENT OUT.PUT
PRINT 1 LINE WITH TIME.V AS FOLLOWS
    EVENT OUT.PUT ENTERED, THE TIME IS ***.**
LIST ATTRIBUTES OF EACH NODE
LIST ATTRIBUTES OF EACH SINK
LIST TREE,CONNECTIVITY
LIST FIL,SFIL
LIST ZIL,DILL,DIL,NIL,SDIL,SNIL
RETURN
END
EVENT STOP.SIMULATION
PRINT 1 LINE WITH TIME.V AS FOLLOWS
    TIME TO STOP, TIME= ***.**
STOP
END
```

APPENDIX B

ALGORITHM FOR NODAL OPERATIONS

Table I - Variables of the Algorithm of Table III.

Variable Name	Meaning	Domain of Values
p_i	preferred neighbor	nil, 1,2,...,K
d_i	estimated distance from SINK	,1,2,3,...
$D_{i\ell}$	estimated distance of link (i, ℓ)	1,2,3,...
n_i	current counter number	0,1,2,...
mx_i	largest number m received by node i	0,1,2,...
CT	control flag	0,1
$N_i(\ell)$	last number m received from ℓ after i completed last update cycle	nil,0,1,2,...
$D_i(\ell)$	$d + d_{i\ell}$ for last d received from ℓ	∞,1,2,...
$F_i(\ell)$	status of link (i, ℓ)	DOWN,READY,UP
$Z_i(\ell)$	synchronization number used by i to bring link (i, ℓ) UP	0,1,2,...

Table II- Messages received by the algorithm of Table 3.

Message Format	Meaning	Domain of Values
MSG(m,d, ℓ)	updating message from ℓ	m = 0,1,2,... d = ∞,0,1,2,... ℓ = 1,2,...,K
FAIL (ℓ)	failures detected on link (i, ℓ)	ℓ = 1,2,...,K
WAKE (ℓ)	link (i, ℓ) becomes operational	ℓ = 1,2,...,K
REQ(m)	request for new update cycle with n_{SINK}	= 0,1,2,...

TABLE III

ALGORITHM FOR NODE i-FINITE-STATE-MACHINE TRANSITIONS

T12	Condition 12	MSG($m=mx_i$, $d \neq \infty$, $l = p_i$), CT=0
	Comment 12	$m \quad n_i$.
	Action 12	$d_i \leftarrow \min_{k:F_i(k)=UP} D_i(k)$ $N_i(k)=m$ $n_i \leftarrow m$; k s.t. $F_i(k)$ = READY if $n_i > z_i(k)$, then $F_i(k) \leftarrow$ UP, $N_i(k) \leftarrow$ nil; transmit MSG(n_i,d_i) to all k s.t. $F_i(k)$= UP and $k \neq p_i$; CT $\leftarrow$ 1.
T13	Condition 13	(MSG($l=p_i$, $d=\infty$, m) or FAIL($l=p_l$)), CT=0.
	Comment 13	If MSG, then $m \quad n_i$.
	Action 13	$d_i \leftarrow \infty$ if MSG, then $n_i \leftarrow m$ k s.t. $F_i(k)$ = READY, if $n_i > z_i(k)$, then $F_i(k) \leftarrow$ UP, $N_i(k) \leftarrow$ nil; transmit MSG(n_i,d_i) to all k s.t. $F_i(k)$= UP and $k \neq p_i$; $p_i \leftarrow$ nil; CT $\leftarrow$ 1.
T21	Condition 21	k s.t.$F_i(k)$ = UP, then $N_i(k)=n_i=mx_i$; k s.t. $F_i(k)$= UP and $D_i(k) \quad d_i$; if CT = 0, then MSG; $D_i(p_i) \neq \infty$
	Comment 21	$d_i \neq \infty$, $p_i \neq$ nil.
	Action 21	Transmit MSG(n_i,d_i) to p_i; $p_i \leftarrow k^*$ that achieves $\min_{k:F_i(k)=UP} D_i(k)$; k s.t.$F_i(k)$ = UP, set $N_i(k) \leftarrow$ nil CT $\leftarrow$ 1.

T22 Conditions 22 $MSG(m=mx_i>n_i,\ d \neq \infty,\ l=p_i), CT=0$

Action 22 Same as Action 12.

T23 Same as T13

T23 Condition 32 k s.t. $F_i(k) = UP$, $mx_i = N_i(k) > n_i$, $D_i(k) \neq \infty$.

Comment 32 $p_i=nil$, $d_i = \infty$.

Action 32 Let k^* achieve $\min_{k:F_i(k)=UP,\ N_i(k)=mx_i} D_i(k)$.

Then $p_i \leftarrow k^*$;

$n_i \leftarrow mx_i$;

$d_i \leftarrow D_i(k^*)$;

k s.t. $F_i(k)$= READY, if $n_i > z_i(k)$, then $F_i(k) \leftarrow$ UP, $N_i(k) \leftarrow$ nil;

transmit $MSG(n_i, d_i)$ to all k s.t. $F_i(k)$= UP and $k \neq p_i$;

$CT \leftarrow 1$.

TABLE IV

ALGORITHM FOR AN ARBITRARY NODE i-MESSAGE HANDLER

For REQ(m)

if $p_i \neq$ nil, then send REQ(m) to p_i.

For FAIL (l)

$F_i(l) \leftarrow$DOWN;

CT $\leftarrow$ 0

Execute FINITE-STATE-MACHINE;

if $p_i \neq$ nil, then send REQ(n_i) to p_i.

For MSG(m,d,l)

if $F_i(l)$ = READY, then $F_i(l) \leftarrow$ UP

(Comment: m> $z_i(l)$);

$N_i(l) \leftarrow$ m;

$D_i(l) \leftarrow$ d + d_{il};

$mx_i \leftarrow$max {m,mx_i;};

CTO$\leftarrow$ 0

Execute FINITE-STATE-MACHINE.

FOR WAKE(l)

(Comment: Assuming $F_i(l)$ = DOWN)

if i and l agree to open link (i,l) then:

$z_i(l) \leftarrow$max {n_i,n_i}

$F_j(l) \leftarrow$ READY;

$N_i(l) \leftarrow$ nil;

if $p_i \neq$ nil, then send REQ($z_i(l)$) to p_i.

TABLE V

THE ALGORITHM FOR THE SINK

For REQ(m)

CT← 0;

Execute FINITE-STATE-MACHINE

For FAIL(l)

$F_i(l) \leftarrow$ DOWN;

CT ← 0;

Execute FINITE-STATE-MACHINE

For MSG(m,d,l)

$N_i(l) \leftarrow m$;

CT ← 0;

Execute FINITE-STATE-MACHINE

For WAKE(l)

(Comment: $F_i(l)$ = DOWN)

if SINK and l agree to open link (SINK,l), then

$F_i(l) \leftarrow$ READY;

CT ← 0;

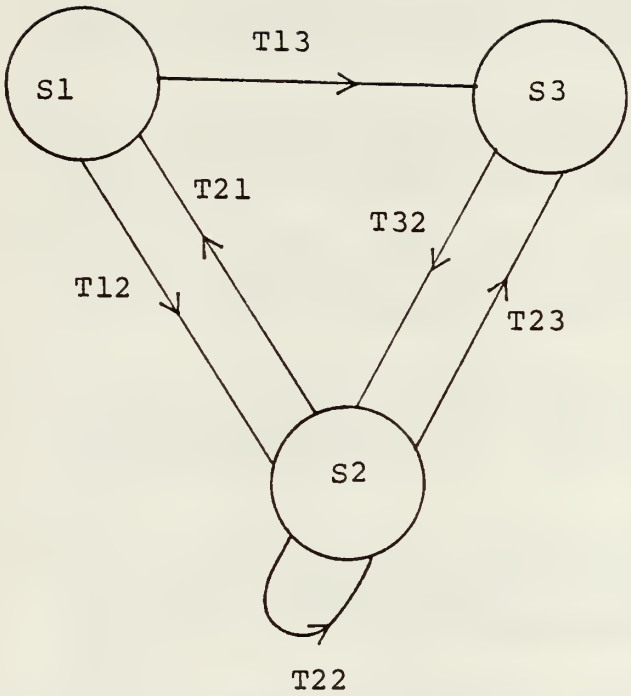
Execute FINITE-STATE-MACHINE.

For START

CT 0;

Execute FINITE-STATE-MACHINE.

T12	Condition 12	(CT=0) and (REQ(m = n_{SINK}) or FAIL or WAKE or START).
	Action 12	if (REQ or FAIL or WAKE), then $n_{SINK} \leftarrow n_{SINK} + 1$; k s.t.$F_i(k)$= READY, then $F_i(k) \leftarrow$ nil; transmit MSG(n_{SINK},0) to all k s.t. $F_i(k)$ = UP;CT ← 1.
T21	Condition 21	k s.t. $F_i(k)$ = UP, then $N_i(k)$ = n_{SINK}; MSG.
	Action 21	k s.t.$F_i(k)$ = UP, then $N_i(k) \leftarrow$ nil. CT 1.
T22	Condition 22	(CT = 0) and (REQ(m = n_{SINK})or FAIL or WAKE)
	Action 22	Same as Action 12.



S = State of Node
T = Allowable Transition

FIGURE 9. Finite State Machine

BIBLIOGRAPHY

1. (ANS 78) "ANSI Reference Model for Distributed Systems" Data Communication Management, pp. 1-10, June 1978.

2. (COM 79) Combined Arms Combat Developments Activity, Ft. Leavenworth Kansas, "Phase I Corps Information Flow (CIF)", 15 April 1979.

3. (CYP 78) Cypser, R.J., Communications Architecture for Distributed Systems, Addison-Wesley, 1978.

4. (DAR 77) Defense Advanced Research Projects Agency Technical Report 4940, Continuous Land Combat by E.J. Emanski Jr., September 1977.

5. (ESL 78) Electronic Systems Laboratory ESL R-846, "JTIDS: An Update" by E.G. Smith, pp. 115-123, September, 1978.

6. (FIN 79) Finn, S.G., "Resynch Procedures and a Fail-safe Network Protocol," IEEE Transactions on Communications, vol. com-27, no. 6, pp. 840-845.

7. (FOU 78) Fouad, T.A., and others, "Modeling and Measurement Techniques in Packet Communication Networks," Proceedings of the IEEE, vol. 66, no. 11, pp. 1423-1447, November, 1978.

8. (GER 79) Gerla, M., Hybrid Switching: A New Distributed Network Technology, paper presented at WESCON -79, San Francisco, California, 9 September 1979.

9. (HEA 79) Headquarters U.S. Army Training and Doctrine Command, Ft. Monroe, VA., unclassified letter ATCD-C-C subject: Letter of Agreement (LOA) for PLRS/JTIDS Hybrid, USATRADOC ACN58401, 6 July 1979.

10. (HAR 72) Harary, F., Graph Theory, Addison-Wesley, 1972.

11. (HUG 79) Hughes Ground Systems Division, MK1 ADDS PLRS Net Management, 1979.

12. (LAW 79) Lawson, B.R., A Testbed for Packet Radio in an Army Tactical Corps, paper presented at EASCON '79, Washington, D.C., 10 October 1979.

13. (MAR 78) Marine Corps Development and Education Command, D035/TCDlss, Landing Force Organizational Systems Study (LFOSS), 1978.

14. (MAR 79) Marine Corps Development and Education Command Developmental Bulletin 1-79, Electronic Warfare Operations Handbook, 5 January 1979.

15. (NAV 78) Naval Ocean Systems Center, San Diego, California, Mobile Command Concept, 5 December 1979.

16. (PRO 78) Proceedings of the IEEE (Special lssal Issue on Packet Radio), vol. 66, no. 11, November 1979.

17. (SID 79) Sidi, M., and Segall A., Failsafe Distributed Routing Protocol, M.S. Thesis, Technion, Haifa Israel, March 1979.

18. (SEG 79) Segall, A., and Merlin, P., "A Failsafe Distributed Routing Protocol," IEEE Transactions on Communications, vol com-27, no. 9, pp xx-xx, September 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 62Ki Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	Professor John M. Wozencraft, Code 74 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
5.	Capt. Edmund A. Lucke, U.S.M.C. 2225 Montgomery Avenue Woodbridge, Virginia 22191	2
6.	Commander Naval Ocean Systems Center Code 033 San Diego, California 92152	2
7.	Dr. A. Segall Technion-Israel Institute of Technology Haifa, Israel	1
8.	Dr. Michael Athans Laboratory for Information and Decision Systems Massachusetts Institute of Technology Boston, Massachusetts 02139	1
9.	Directorate of Combat Developments JINTACCS (Capt. Hitchcock) USAC, Ft. Gordon, Georgia 30905	1
10.	Marine Corps Development and Education Command C3 Division, Development Center Systems Definition Branch (Lt Col Bronson) Quantico, Virginia 22134	1
11.	Marine Corps Base Camp Pendleton MCTSSA (Code LIFICS) Camp Pendleton, California 92055	1

12. Lt. Ellen Roland U.S.N. (Code 55Ro) — 1
Operations Research Department
Naval Postgraduate School
Monterey, California 93940

13. Defense Advanced Research Projects Agency — 1
Information Processing Techniques Office
ATTN: Col. Hammett
Washington, D.C. 20390

14. Naval Electronics Systems Command — 1
Code 03
Washington, D.C. 20390

15. Naval Electronics Systems Command — 1
Code 540
Washington, D.C. 20390

16. Naval Material Command — 1
Code 08TM (Lt.Col. Bowles, USMC)
Washington, D.C. 20390

17. Office of Naval Research — 1
Code 100M (Col. Clark USMC)
Washington, D.C. 20390

18. Commander Naval Ocean Systems Center — 3
Code 8105
San Diego, California 92152

19. Commander Naval Ocean Systems Center — 1
Code 814 (Jerry Clapp)
San Diego, California 92152

20. Commander Naval Ocean Systems Center — 2
Code 447 (Library)
San Diego, California 92152

21. Professor H. Titus (Code 62Ts) — 1
Electrical Engineering Department
Naval Postgraduate School
Monterey, California 93940

22. Professor D.P. Gaver (Code 55Gv) — 1
Operations Research Department
Naval Postgraduate School
Monterey, California 93940

Thesis
L8917 Lucke
c.1

186494

An investigation of distributed communications systems and their potential applications to the command and control structure of the Marine Corps.